Jardin A

Clase Jardin, es un hilo

```
import java.util.ArrayList;

public class Jardin extends Thread {

    public static int personas= 100;
    public static boolean puertaEntrada=true;
    public static boolean puertaSalida=true;
    public ArrayList<Persona> listaPersonas= new ArrayList<>();
    public static int fallos=0;

public Jardin (){
    super.run();
}
    @Override
    public void run() {
    for (Persona persona: listaPersonas){
        persona.start();
    }
        try {
            Thread.sleep(10000);
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
        for (Persona persona: listaPersonas){
            persona.interrupt();
        }
        System.out.println("fallos: "+fallos);
    }
}
```

Clase persona y Enum estado, con valores fuera y dentro

```
enum Estado{dentro,fuera}

public class Persona extends Thread {
   public int numPersona;
   public Estado situacion;
   public Persona(int numPersona, Estado estado) {
      super();
      this.numPersona = numPersona;
      this.situacion = estado;
   }

   @Override
   public void run() {
      while (true) {
         synchronized (this) {
            if (situacion == Estado.dentro) {
               if (Jardin.puertaSalida) {
                  if (Jardin.personas>0){
                  Jardin.puertaSalida = false;
                  System.out.println("saliendo " + numPersona);
                  try {
                     Thread.sleep(3000);
                  } catch (InterruptedException e) {
                     return;
                  }
                  System.out.println("salio " + numPersona);
                  Jardin.personas--;
                  situacion = Estado.fuera;
                  Jardin.puertaSalida = true;
                  } else {
                        System.out.println(" no sali, no habia nadie " + numPersona);
                        Jardin.fallos += 1;
                        try {
                           Thread.sleep(1000);
                        } catch (InterruptedException e) {
                           return;
                        }
                     }
               }else {
                  System.out.println(" no sali, puerta ocupada " + numPersona);
                  Jardin.fallos += 1;
                  try {
                     Thread.sleep(1000);
                  } catch (InterruptedException e) {
                     return;
                  }
               }
            }
         }
         synchronized (this) {
            if (situacion == Estado.fuera) {
               if (Jardin.puertaEntrada) {
                  Jardin.puertaEntrada = false;
                  System.out.println("entrando " + numPersona);
                  try {
                     Thread.sleep(3000);
                  } catch (InterruptedException e) {
                     return;
```

```
                    }
                    System.out.println("entrado " + numPersona);
                    Jardin.personas++;
                    situacion = Estado.dentro;
                    Jardin.puertaEntrada = true;

                } else {
                    System.out.println(" no entre, puerta ocupada " + numPersona);
                    Jardin.fallos += 1;
                    try {
                        Thread.sleep(1000);
                    } catch (InterruptedException e) {
                        return;
                    }
                }
            }
        }
    }
}
```

Clase main

```
import java.util.Scanner;

//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or
// click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.
public class Main {
    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
        int dentro=0;
        int fuera=0;
        int numero=0;
        System.out.println("personas dentro");
        dentro=in.nextInt();
        in.nextLine();

        System.out.println("personas fuera");
        fuera=in.nextInt();
        in.nextLine();
        Jardin jar=new Jardin();

        for (int z=0;z<dentro;z++){
            Persona per1=new Persona(numero,Estado.dentro);
            jar.listaPersonas.add(per1);
            numero++;
        }

        for (int x=0;x<fuera;x++){
            Persona per2=new Persona(numero,Estado.fuera);
            jar.listaPersonas.add(per2);
            numero++;
        }

        jar.start();

    }
}
```

Resultado

```
"C:\Users\a24borjapp\AppData\Local\Programs\Eclipse Adoptium\jdk-21.0.8.9-hotspot\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\Inte
personas dentro
4
personas fuera
4
```

```
 no sali, puerta ocupada 4
entrando 6
 no entre, puerta ocupada 5
 no sali, puerta ocupada 1
salio 2
 no entre, puerta ocupada 2
 no sali, puerta ocupada 3
 no entre, puerta ocupada 7
saliendo 0
 no sali, puerta ocupada 4
 no sali, puerta ocupada 3
 no entre, puerta ocupada 2
 no sali, puerta ocupada 1
 no entre, puerta ocupada 5
 no entre, puerta ocupada 7
 no sali, puerta ocupada 4
 no entre, puerta ocupada 5
 no sali, puerta ocupada 1
 no entre, puerta ocupada 2
 no sali, puerta ocupada 3
 no entre, puerta ocupada 7
entrado 6
 no sali, puerta ocupada 6
 no sali, puerta ocupada 4
 no sali, puerta ocupada 3
 no entre, puerta ocupada 2
 no sali, puerta ocupada 1
entrando 5
 no entre, puerta ocupada 7
fallos: 61

Process finished with exit code 0
```

Jardin B

Clase Jardin

```
import java.util.ArrayList;

public class Jardin extends Thread {

    public static int personas;
    public static int personasMax;
    public static ArrayList<Boolean>puertas=new ArrayList<>();
    public ArrayList<Persona> listaPersonas= new ArrayList<>();
    public static int fallos=0;

public Jardin (ArrayList<Boolean> puertas,ArrayList<Persona> listaPersonas,int personasMax,int personas){
    super.run();
    this.listaPersonas=listaPersonas;
    this.personas=personas;
    this.puertas=puertas;
    this.personasMax=personasMax;
}
    @Override
    public void run() {
    for (Persona persona: listaPersonas){
        persona.start();
    }
        try {
            Thread.sleep(10000);
        } catch (InterruptedException e) {}

        for (Persona persona: listaPersonas){
          persona.interrupt();
        }
        System.out.println("intentos fallidos: "+fallos);
    }

    public static void setXpuertaFalse(int i){
    puertas.set(i,false);
    }

    public static void setXpuertaTrue(int i){
        puertas.set(i,true);
    }
    public static void addFallos(){
      fallos++;
    }
}
```

Clase persona y enum de estados

```
import java.util.Random;

enum Estado{dentro,fuera}

public class Persona extends Thread {
    public int numPersona;
    public Estado situacion;
    public Boolean funcionando=true;
    public Random random=new Random();
    public Persona(int numPersona, Estado estado) {
        super();
        this.numPersona = numPersona;
        this.situacion = estado;
    }

    @Override
    public void run() {
        while (funcionando) {
            synchronized (this) {
                int i = random.nextInt(Jardin.puertas.size());
                if (situacion == Estado.dentro) {
                    if (Jardin.puertas.get(i)) {
                        if (Jardin.personas >0) {
                            Jardin.setXpuertaFalse(i);
                            System.out.println("saliendo puerta " + (i + 1) + " persona " + numPersona);
                            try {
                                Thread.sleep(3000);
                            } catch (InterruptedException e) {
                                return;
                            }
                            System.out.println("salio puerta " + (i + 1) + " persona " + numPersona);
                            Jardin.personas--;
                            situacion = Estado.fuera;
                            Jardin.setXpuertaTrue(i);
                        } else {
                            System.out.println("no habia nadie para salir " + (i + 1) + " persona " + numPersona);
                            Jardin.addFallos();
                            try {
                                Thread.sleep(1000);
                            } catch (InterruptedException e) {
                                return;
                            }
                        }

                    } else {
                        System.out.println(" no sali puerta " + (i + 1) + " , puerta ocupada " + numPersona);
                        Jardin.addFallos();
                        try {
                            Thread.sleep(1000);
                        } catch (InterruptedException e) {
                            return;
                        }
                    }

                }
            }

            synchronized (this) {
                int i = random.nextInt(Jardin.puertas.size());
                if (situacion == Estado.fuera) {
                    if (Jardin.puertas.get(i)) {
```

```
                if (Jardin.personas < Jardin.personasMax) {
                    Jardin.setXpuertaFalse(i);
                    System.out.println("entrando puerta " + (i + 1) + " persona " + numPersona);
                    try {
                        Thread.sleep(3000);
                    } catch (InterruptedException e) {
                        return;
                    }

                    System.out.println("entrado puerta " + (i + 1) + " persona " + numPersona);
                    Jardin.setXpuertaTrue(i);
                    Jardin.personas++;
                    situacion = Estado.dentro;
                } else {
                    System.out.println(" no entre puerta " + (i + 1) + " , estaba lleno " + numPersona);
                    Jardin.addFallos();
                    try {
                        Thread.sleep(1000);
                    } catch (InterruptedException e) {
                        return;
                    }
                }
            } else {
                System.out.println(" no entre puerta " + (i + 1) + " , puerta ocupada " + numPersona);
                Jardin.addFallos();
                try {
                    Thread.sleep(1000);
                } catch (InterruptedException e) {
                    return;
                }
            }
        }
    }
}

    public void setFuncionandoFalse(){
    this.funcionando=false;
    }
}
```

Clase main

```
import java.util.ArrayList;
import java.util.Scanner;

//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or
// click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.
public class Main {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);
        System.out.println("introduce numero de puertas 1-4");
        int puertas=input.nextInt();
        input.nextLine();
        ArrayList<Boolean> puertasCrear=new ArrayList<>();
        for (int i=0;i<puertas;i++)puertasCrear.add(true);

        int personasMax=-1;
            System.out.println("introduce numero de personas maximas 1-999");
            personasMax= input.nextInt();

        int personas=-1;
```

```
        do {
            System.out.println("introduce numero de personas 1-100");
            personas=input.nextInt();
        }while (personas>personasMax || personas<0|| personas>100);

        ArrayList<Persona> personasCrear=new ArrayList<>();
        for(int z=0;z<personasMax;z++)personasCrear.add(new Persona(z,Estado.fuera));

        Jardin jar=new Jardin(puertasCrear,personasCrear,personasMax,personas);
        jar.start();

    }
}
```

Resultado

```
"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\Users\elgoc\AppData\Local\JetBrains\IntelliJ IDEA Community Edition 2025.1.2\lib\idea_rt.jar=54856" -Dfile.encod
introduce numero de puertas 1-4
2
introduce numero de personas maximas 1-999
4
introduce numero de personas 1-100
2
entrando puerta 2 persona 0
```

```
 entrado puerta 1 persona 2
 saliendo puerta 2 persona 0
  no sali puerta 2 , puerta ocupada 2
  no entre puerta 2 , puerta ocupada 1
  no entre puerta 2 , puerta ocupada 3
  no sali puerta 2 , puerta ocupada 2
  no entre puerta 1 , estaba lleno 1
  no entre puerta 2 , puerta ocupada 3
  no entre puerta 1 , estaba lleno 1
  no entre puerta 1 , estaba lleno 3
 salio puerta 2 persona 0
 entrando puerta 2 persona 0
  no entre puerta 2 , puerta ocupada 1
 saliendo puerta 1 persona 2
  no entre puerta 1 , puerta ocupada 3
  no entre puerta 2 , puerta ocupada 1
  no entre puerta 1 , puerta ocupada 3
  no entre puerta 2 , puerta ocupada 1
  no entre puerta 2 , puerta ocupada 3
 entrado puerta 2 persona 0
  no entre puerta 2 , estaba lleno 1
 salio puerta 1 persona 2
 entrando puerta 1 persona 2
 entrando puerta 2 persona 3
 intentos fallidos: 23

 Process finished with exit code 0
```