# Prácticas de Laboratorio. SINT

## Amueblando el cerebro: búsquedas

Universidad de Huelva

# ENTREGA

Fecha de entrega:

**21 de Mayo de 2022 23:59:59**

Método de entrega:

**Tarea en Moodle**

Entregable (.zip):

**Código fuente: no evaluable - obligatoria la entrega - base de las siguientes**

# ENTREGA

Siguientes prácticas…

STRIPS y SUDOKU

**Las entregas se permitirán hasta el día antes del examen: martes 14 de Junio.**

# OBJETIVOS GENERALES

En la práctica 2 aprendimos a:

**HEMOS AMUEBLDO EL CEREBRO**

En esta práctica tendremos como objetivo inicial:

**CAMBIAR EL RAZONAMIENTO: búsquedas**

Como objetivo principal:

**aplicar un *razonamiento basado en búsqueda en el espacio de estados A****

# OBJETIVOS ESPECÍFICOS I

**Objetivo 1**:

Representación del entorno: YA LO TENEMOS

**Objetivo 2**:

creación de listas de objetos, asociados por su tipología: YA LO TENEMOS

# OBJETIVOS ESPECÍFICOS II

**Objetivo 3**:

Creación de un motor que devuelva el camino/acción hacia la salida.

Para ello desarrollaremos el algoritmo A* como estrategia para buscar el camino hacia la solución.

**Dar 1 paso (mejor) y calcular el camino restante hasta el final**

# OBJETIVOS ESPECÍFICOS II

BUSQUEDA ANCHURA

```
1  public Lista<Accion> Busqueda(Nodo inicial,fin) {
2      Lista Abiertos = new Lista();
3      Lista Cerrados = new Lista();
4      Abiertos.Insertar(inicial);
5      while (Abiertos.Tamano()>0) {
6          Nodo Actual = Abiertos.SacarPrimero();
7          Cerrados.Insertar(Actual);
8          if (Actual == fin) {
9              return Recuperar_Camino(inic,Actual);
10         }
11         else {
12             List<Nodo> sucesores = getSucesores(Actual);
13             sucesores = QuitarRepetidos(sucesores,Cerrados);
14             Abiertos.Insertar_AL_FINAL(sucesores);
15         }
16     }
17     return null;
18 }
```

# EJERCICIOS

**EJERCICIO 1: (lo tenemos de las prácticas iniciales)**

Seleccionar el juego 50

Crear una clase Mundo que contenga la representación del tablero VACIO y un conjunto de listas con los OBJETOS del mapa.

Crear un método que lo pinte para verificar que la interpretación es correcta. Pintar las diferentes categorías…

Una vez comprobada, se puede obviar el método.

# EJERCICIOS

**Ejercicio 2:**

Crear el razonamiento (cerebro) basado en la búsqueda en el espacio de estados utilizando A*.

Se tendrá que resolver el juego **16**

# ANOTACIONES

**Anotaciones**

Nuestro nuevo motor ***BEspacioEstados*** (extienda o no de razonador) debe implementar el método piensa(), que contendrá el algoritmo de búsqueda y los métodos necesarios para su implementación.

Al método habrá que pasarle el mundo y los operadores posibles

**¿cómo quedaría nuestro agente?**

# ANOTACIONES

**Anotaciones**

¿Dónde pensamos?

A:

B:

¿POR QUÉ?

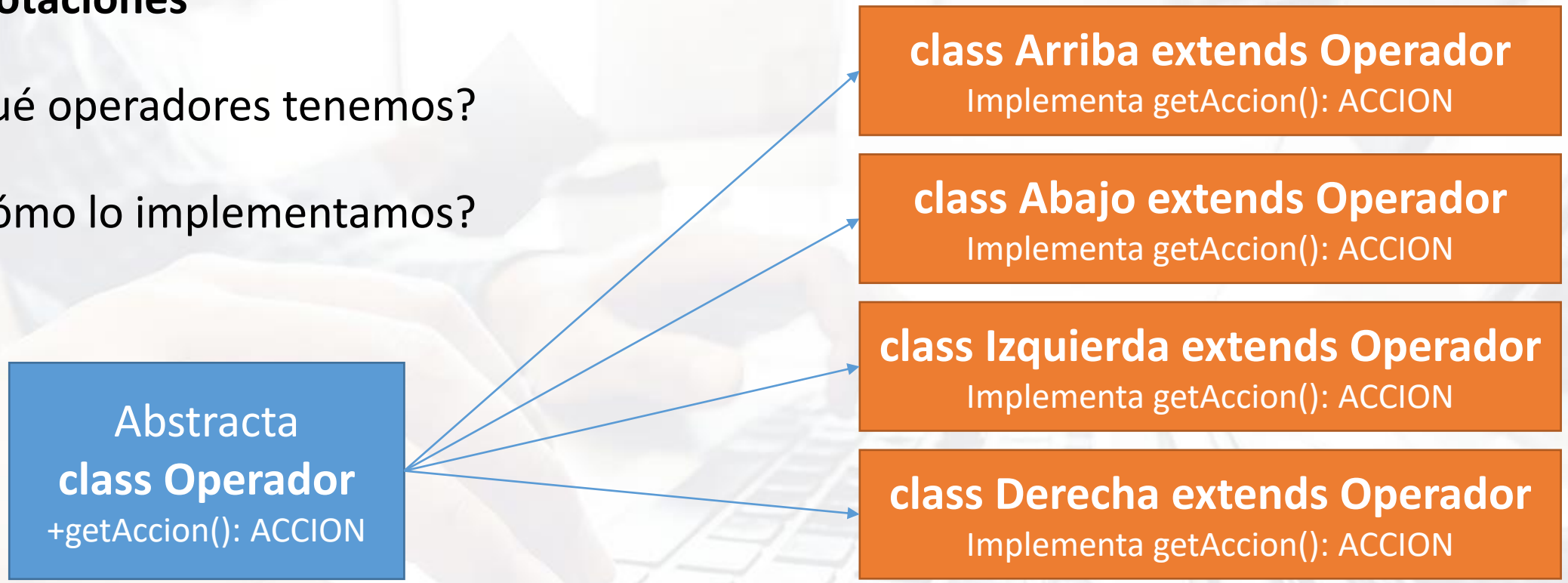
```
10  import generico.interfaces.Operador;
11  import ontology.Types.ACTIONS;
12  import tools.ElapsedCpuTimer;
13  import problema.acciones.*;
14  import problema.modelo50.Mundo50;
15
16  public class Practica_05_50 extends AbstractPlayer {
17
18      // Atributos persistentes del agente
19
20      BEspacioEstados cerebro;
21      Mundo50 mundo;
22
23      List<Operador> solucion;
24      List<Operador> acciones;
25      int contador = 0;
26
27      public Practica_05_50(StateObservation stateObs, ElapsedCpuTimer elapsedTimer)
28      {
29          //calcular solucion??
30      }
31
32      public ACTIONS act(StateObservation stateObs, ElapsedCpuTimer elapsedTimer) {
33
34          ACTIONS a;
35          //calcular solucion??
36          return a;
37      }
38  }
```

# ANOTACIONES

**Anotaciones**

¿qué operadores tenemos?

¿Cómo lo implementamos?



# ANOTACIONES

**Anotaciones**

¿Qué es un estado?

¿Qué debe hacer como mínimo?

**interface Estado**
+aplicarOperador(Operqdor o): Estado
+isMeta(): bool
+actualizar(StateObservation o): void

**class Mundo50 implements Estado**
Implementa +aplicarOperador(Operador o): Estado
implementa +isMeta(): bool
Implementa +actualizar(StateObservation o): void

# ANOTACIONES

**Anotaciones**

¿Cómo hacemos recuperar camino cuando encontramos meta: actual == fin?

```
mientras (nodoactual.obtenerPadre() != null) {

        camino.añadir(nodoactual.getOperador());

        nodoactual = nodoactual.obtenerPadre();

}
```

# EJEMPLO DE SOLUCIÓN

# MAPA OBJETIVO

**JUEGO 16**

*Movimiento de traslación*

*Pared muere*

*Agua muere*

*¿Cómo se calcula el siguiente estado?*