# Práctica 3 IA – Planificación de menús semanales (PDDL)

Este repositorio implementa el modelado PDDL y los casos de prueba para la práctica de planificación. Se cubren **todas las extensiones (0 → 5)** y se incluye el **punto extra**: un generador automático de problemas.

## Jerarquía del proyecto

```
prac3IA\_PDDL/
├── .gitignore
├── README.md
│
├── domain.pddl            ← Dominio STRIPS + typing (niveles 0-3)
├── domain\_metric.pddl      ← Igual que acima + funciones y \:metric (niv. 4-5)
│
├── problems/              ← SOLO archivos de problema (.pddl)
│   ├── basic/
│   │   ├ basic01.pddl
│   │   └ basic02.pddl
│   ├── ext1\
│   ├── ext2\
│   ├── ext3\
│   ├── ext4\
│   └── ext5\
│       └ …                ← cada carpeta tendrá ≥2 ficheros generados
│
├── tools/
│   ├── gen\_problems.py     ← Genera problemas aleatorios (punto extra)
│   └── run\_all.sh          ← Corre problemas/ y ejecuta los planificadores
│
```

## Requisitos del entorno

| Herramienta | Uso |
|---|---|
| **Fast-Forward v2.3** (`ff.exe`) | Extensiones 0 → 3 |
| **Metric-FF** (`metricff.exe`) | Extensiones 4 y 5 |
| **Python 3.x** | Ejecutar `tools/gen_problems.py` |
| Bash / PowerShell | Lanzar scripts y planificadores |

> Coloca `ff.exe`, `metricff.exe` y `cygwin1.dll` en una carpeta de tu `PATH` (por ejemplo `C:\PDDL\planificadores\`).

---

## 🚀 Cómo generar y ejecutar casos

### 1. Generar problemas (punto extra)

```
# Ejemplo: crear 2 problemas para la extensión 2
python3 tools/gen_problems.py \
    --extension ext2_no_type_repeat \
    --count 2 \
    --output problems/ext2_no_type_repeat/
```

El script añade los ficheros `ext2_01.pddl`, `ext2_02.pddl`...

---

### 2. Ejecutar un problema aislado

**Niveles básicos (0-3):**

```
ff.exe -o domain.pddl -f problems/basic/basic01.pddl
```

**Extensiones con métricas (4-5):**

```
metricff.exe -o domain_metric.pddl -f problems/ext5_min_cost/ext5_01.pddl
```

---

### 3. Ejecutar todos los tests

```
bash tools/run_all.sh
```