INF-253 Lenguajes de Programación
Tarea 5 2025-2: Malla Interactiva

Profesores: José Luis Martí Lara, Wladimir Ormazábal Orellana, Ricardo Salas Letelier
Ayudante coordinador: Bastián Ríos Lastarria
Ayudantes: Diego Duarte, Andrés Jablonca, Cristobal Tirado,
Martin Pardo, Carlos Bravo, Martín Palacios,
Blas Olivares, Sofía Ramirez, Benjamín Echeverria

14 de noviembre de 2025

Índice

1. Introducción

El Departamento de Informática (DI) está haciendo unos reajustes a la malla curricular y necesita realizar varias consultas y ver cómo se comporta la misma en diversos casos. El DI, interesado por tus anteriores proyectos, te da la siguiente misión: como estudiante de la carrera, se te asigna recrear la malla curricular que estás cursando actualmente como un **grafo dirigido** en el lenguaje de programación `Prolog` y crear predicados que permita realizar las consultas necesarias. Cada asignatura es un nodo y cada prerrequisito es una arista desde el requisito hacia el asignatura que habilita.
El programa que se utilizará para revisar y evaluar la tarea será **SWI-Prolog**. Es altamente recomendado que utilicen el mismo programa para no tener problemas de compatibilidad.

Simbología. Para cada predicado, usaremos `+` para argumentos de entrada ya instanciados y `-` para salidas que el predicado calculará (salida).

2. Funciones a implementar

En esta entrega se evalúan los siguientes **hechos** y **predicados** que deberán ser incluidos en el archivo '`malla.pl`'.

Hechos obligatorios en `malla.pl`

- `asignatura(Codigo, Nombre, Creditos, SemestrePlan, Orden).` ($SemestrePlan \in \{1, \ldots, 11\}$)
- `prerrequisitos(Asignatura, Requisitos).` (*Requisitos* debe aprobarse antes de *Asignatura*)

```
% asignatura(Codigo, Nombre, Creditos, SemestrePlan, Orden).
asignatura('IWI-131', 'Programacion', 5, 1, 1).
asignatura('INF-134', 'Estructuras de Datos', 6, 3, 12).
asignatura('INF-239', 'Bases de Datos', 5, 5, 24).

% prerrequisitos(Asignatura, Requisitos).
prerrequisitos('INF-134', ['IWI-131']).
prerrequisitos('INF-236', ['INF-253','INF-239']).
```

Para evitar problemas de coherencia, se les solicitará que los hechos de asignatura y prerrequisito sean extraídos de la malla interactiva de Ingeniería Civíl Informática existente en mallas.labcomp.cl. *SemestrePlan* corresponde al semestre dentro de la malla curricular y *Orden* es el número identificador asociado a cada asignatura según la información de la página anteriormente dada.

3. Predicados

3.1. habilitada/2

Firma: `habilitada(+Asignatura, +Aprobados)`
Salida: Retorna verdadero si `Asignatura` *no* está en `Aprobados` y *todos* sus prerrequisitos sí lo están.
Ejemplos de uso:

```
?- habilitada('INF-134', ['IWI-131']).
true.

?- habilitada('INF-239', ['IWI-131']).     % falta INF-134
false.

?- habilitada('INF-239', ['IWI-131','INF-134']).
true.
```

3.2. es_prerrequisito/2

Firma: `es_prerrequisito(+Pre, +Asignatura)`
Salida: Verdadero si `Pre` es prerrequisito *directo o indirecto* de `asignatura` (cierre transitivo).
Ejemplos de uso:

```
?- es_prerrequisito('IWI-131', 'INF-134').  % directo
true.

?- es_prerrequisito('IWI-131', 'INF-239').  % por cadena IWI-131 -> INF-134 -> INF-239
true.
```

3.3. permite_dar/3

Firma: `permite_dar(+AprobadosPrev, +AsignaturaRecienAprobada, -NuevosHabilitados)`
Salida: `NuevosHabilitados` es la lista (sin duplicados) de asignaturas que *pasan a quedar habilitados* al agregar `AsignaturaRecienAprobada` al conjunto `AprobadosPrev`, y que *no* estaban habilitados antes.
Ejemplos de uso:

```
?- permite_dar(['IWI-131'], 'INF-134', L).
L = ['INF-253','INF-239','INF-245'].

?- permite_dar(['IWI-131','INF-134'], 'INF-239', L).
L = [].  % nada nuevo que dependa exclusivamente de INF-239
```

3.4. siguiente_semestre/4

Firma: `siguiente_semestre(+Aprobados, +SemActual, +MaxCredSem, -AsignaturasSugeridas)`
Salida: `AsignaturasSugeridas` es la **lista óptima para el siguiente semestre** bajo tope `MaxCredSem`. La selección *siempre* prioriza (1) asignaturas de **semestres más antiguos pendientes** $(1, 2, \ldots,$ `SemActual`$)$, luego los del **mismo semestre** y, sólo si queda cupo, continúa

con **semestres posteriores**. Dentro de esa prioridad, el algoritmo elige un subconjunto cuya **suma de créditos** sea $\leq$ `MaxCredSem` y **máxima posible**.
Desempates (orden estable): primero por `SemestrePlan` ascendente y luego por `Orden` ascendente (número identificador).
Supuestos: toda asignatura se dicta cada semestre (no existe par/impar), sin correquisitos y el grafo es acíclico.
Ejemplos de uso:

```
% Caso 1: Inicio de carrera con 35 creditos: todo 1er semestre y sobran creditos para
    agregar parte del 2do semestre.
?- siguiente_semestre([], 1, 35, L).
L = ['IWI-131','MAT-021','FIS-100','HRW-132','DEW-100','QUI-010','IWG-101','HRW-133'].

% Caso 2: Mismo escenario pero tope 24: solo los del 1er semestre (24 exactos).
?- siguiente_semestre([], 1, 24, L).
L = ['IWI-131','MAT-021','FIS-100','HRW-132','DEW-100'].

% Caso 3: Arrastre pendiente de semestres anteriores tiene prioridad.
?- siguiente_semestre(['IWI-131','MAT-021'], 2, 18, L).
L = ['FIS-100','HRW-132','DEW-100','MAT-022'].  % completa 1er semestre antes de 2do,
    sobran 7 creditos que toma MAT-022
```

3.5. titulacion_optima/4

Firma: `titulacion_optima(+Aprobados, +SemActual, +MaxCredSem, -Plan)`
Salida: `Plan` es la **secuencia óptima de semestres** para completar toda la malla curricular, partiendo en `SemActual`, bajo un tope de créditos `MaxCredSem` por semestre. En cada paso se calcula el *siguiente plan de acción óptimo* respetando prerrequisitos y prioridades, y se avanza hasta que no queden asignaturas pendientes.
Prioridades y desempates (por *cada* semestre): Se debe realizar el mismo órden de prioridades y criterios de desempates definidos en el anterior predicado `siguiente_semestre/4`, aplicado por cada semestre.
Ejemplos de uso:

```
% Ej: Desde el inicio de carrera con tope de 32 creditos.
?- titulacion_optima([], 1, 32, Plan).
Plan = [ sem(1, ['IWI-131','MAT-021','FIS-100','HRW-132','DEW-100','QUI-010','IWG-101'
    ]),
         sem(2, ['MAT-022','FIS-110','HRW-133','DEW-101','INF-134','INF-152','INF-1']),
         sem(3, ['MAT-023','FIS-130','INF-260','INF-253','INF-155','INF-2']),
          sem(4, ['MAT-024','FIS-120','IWN-170','INF-239','INF-245','INF-3']),
          sem(5, ['FIS-140','INF-280','INF-270','INF-236','INF-246'INF-4','INF-5']),
          sem(6, ['INF-276','INF-221','INF-292','INF-225','INF-256','ICN-270','INF-6']),
          sem(7, ['INF-285','INF-293','INF-322','INF-343','INF-305','INF-295']),
          sem(8, ['INF-266','INF-306','INF-307','INF-311','INF-312','INF-7']),
          sem(9, ['INF-360','INF-308','INF-313','INF-314']),
          sem(10, ['INF-228','INF-309']), sem(11, ['INF-310']).
%   Se puede observar con este ejemplo que a un flujo constante de 32 creditos sin
    reprobar ninguna asignatura, muchos semestres quedan con holgura permitiendo
    adelantar ramos, y los ultimos semestres solo se ven limitados por prerrequisitos!
```

Notas: `siguiente_semestre/4` es un componente clave, con el anterior predicado la implementación de éste se vuelve más sencilla!

4. Sobre la entrega

- El código debe venir indentado y ordenado.
- Se debe entregar los siguientes archivos:
  - malla.pl
  - README.txt
- ▶ **El Uso de IA está prohibido**. Si se detecta, se informará a las respectivas autoridades sin derecho a apelación. (aplica para la totalidad de la entrega)
- Si no existe orden en el código habrá descuento.
- Todos los predicados y hechos adicionales deben ir comentados, formato libre, debe incluir nombre del predicado/hecho, comportamiento general e información sobre las entradas y salida. **Se harán descuentos por predicado/hecho no comentado**
- La entrega debe realizarse en un archivo comprimido en tar.gz y debe llevar el nombre: Tarea5LP_RolAlumno.tar.gz.
  Ej.: Tarea5LP_202473000-k.tar.gz.
- El archivo README.txt debe contener el nombre y rol del alumno e instrucciones detalladas para la correcta utilización de su programa.
- La entrega será vía aula y el plazo máximo de entrega es hasta el día **Miércoles 19 de Noviembre, 23:59 horas, vía aula.**
- Por cada día de atraso se descontarán 20 puntos (10 puntos dentro de la primera hora)
- Las copias y Uso de código generado por IA serán informadas a las respectivas autoridades.
- Solo se pueden realizar consultas respecto a la tarea hasta 2 días antes de la entrega.
- Las consultas del enunciado se debe realizarse mediante el foro de la tarea disponible en AULA.

5. Calificación

5.1. Entrega

Para la calificación de su tarea, debe realizar una entrega con requerimientos mínimos que otorgarán 30 pts base, luego se entregará puntaje dependiendo de los otros requerimientos que llegue a cumplir.

5.1.1. Entrega Mínima (total 30 pts)

- Incluir correctamente los hechos obligatorios en ***malla.pl*** *asignatura* y *prerrequisitos* tal y como se presentan en la malla curricular. (10 pts)
- Implementar correctamente el predicado *habilitada/2*. (10 pts)
- Implementar correctamente el predicado *es_prerrequisito/2*. (10 pts)

De obtener los 30 puntos de la entrega mínima, se procederá a la revisión de las siguientes secciones de la tarea.

5.1.2. Asignación de puntajes general (total 70 pts)

- Implementar correctamente el predicado *permite_dar/3*. (20 pts)
- Implementar correctamente el predicado *siguiente_semestre/4*. (30 pts)
- Implementar correctamente el predicado *titulacion_optima/4*. (20 pts)

5.2. Descuentos

- Falta de comentarios sobre cada predicado (–5 pts c/u, máx. –20 pts)
- Código no compila (-100 pts)
- Falta de README (–20 pts)
- Falta de información obligatoria en el README (nombre, rol, instrucciones) (–5 pts c/u)
- Falta de orden o mala indentación en el código (–5 a –20 pts según gravedad)
- Mal nombre en los archivos entregados (–5 pts c/u; –10 pts si es el `.tar.gz`)
- Uso de código/texto generado por IA será informado a las respectivas autoridades.
- Entrega tardía (–10 pts si es dentro de la primera hora; –20 pts por cada día o fracción de retraso)
- En caso de existir nota negativa, esta será remplazada por un 0.