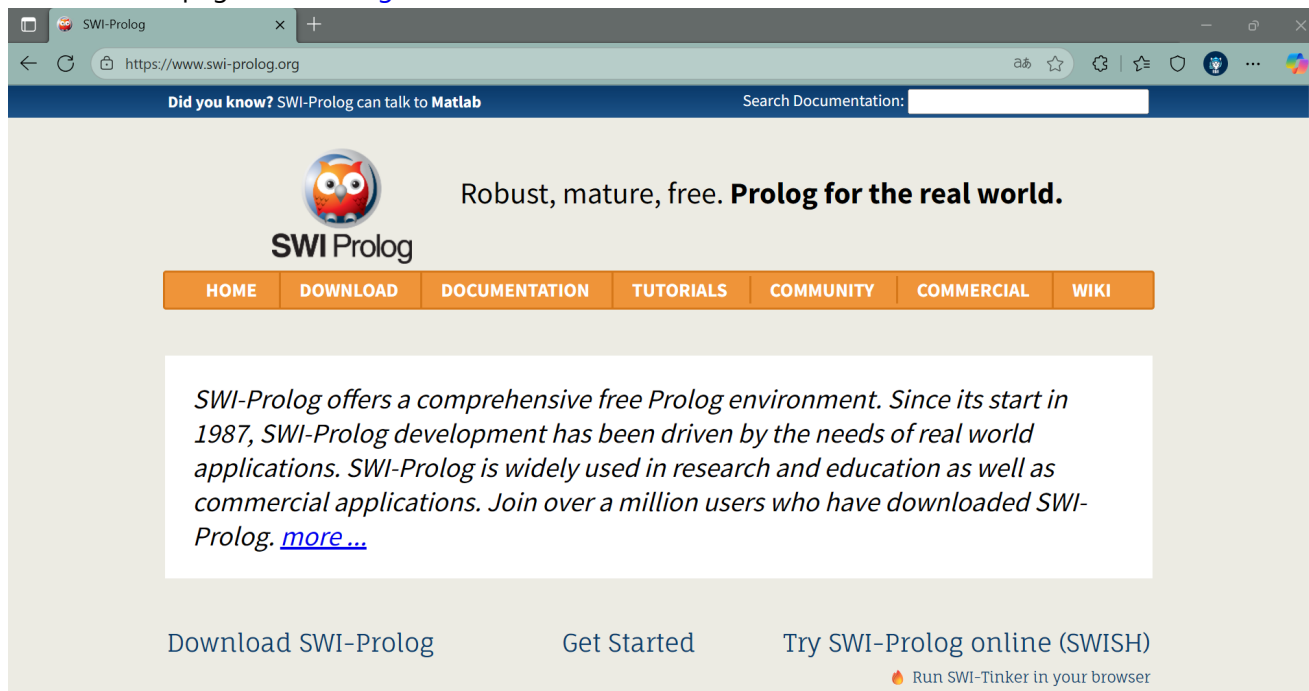


# Práctica 4: Prolog

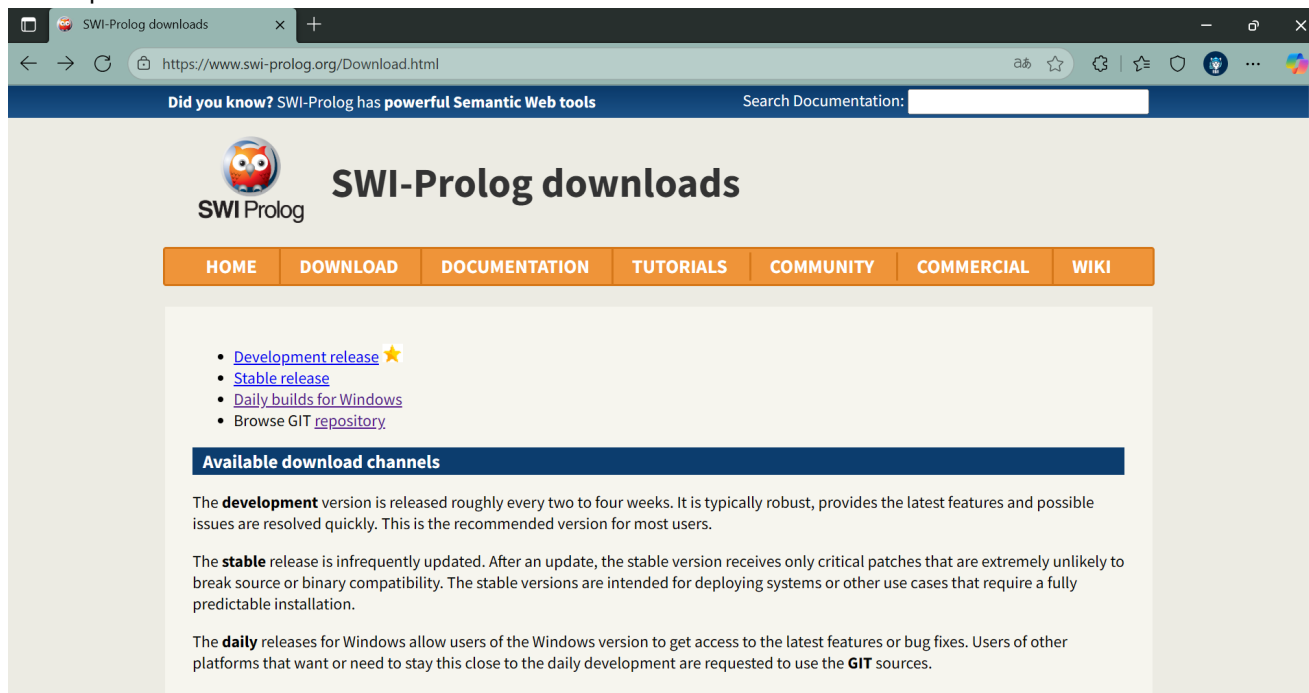
Isidro Francisco Pérez Paz - 377806

## Instalación del entorno de desarrollo

Entramos a la página de [Prolog](https://www.swi-prolog.org).

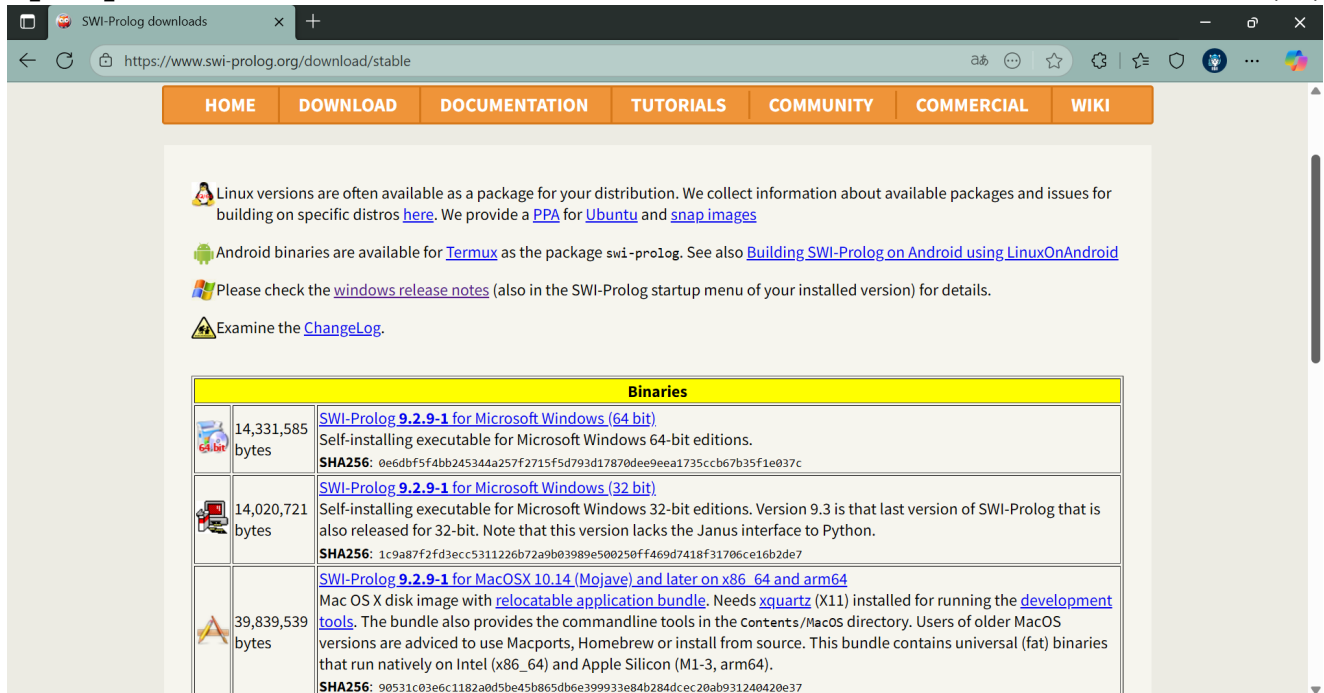


Ve al apartado de [Download](#).






Te vas a la opción de [Stable release](#).

- Selecciona la versión que más se adapte a tu sistema operativo.

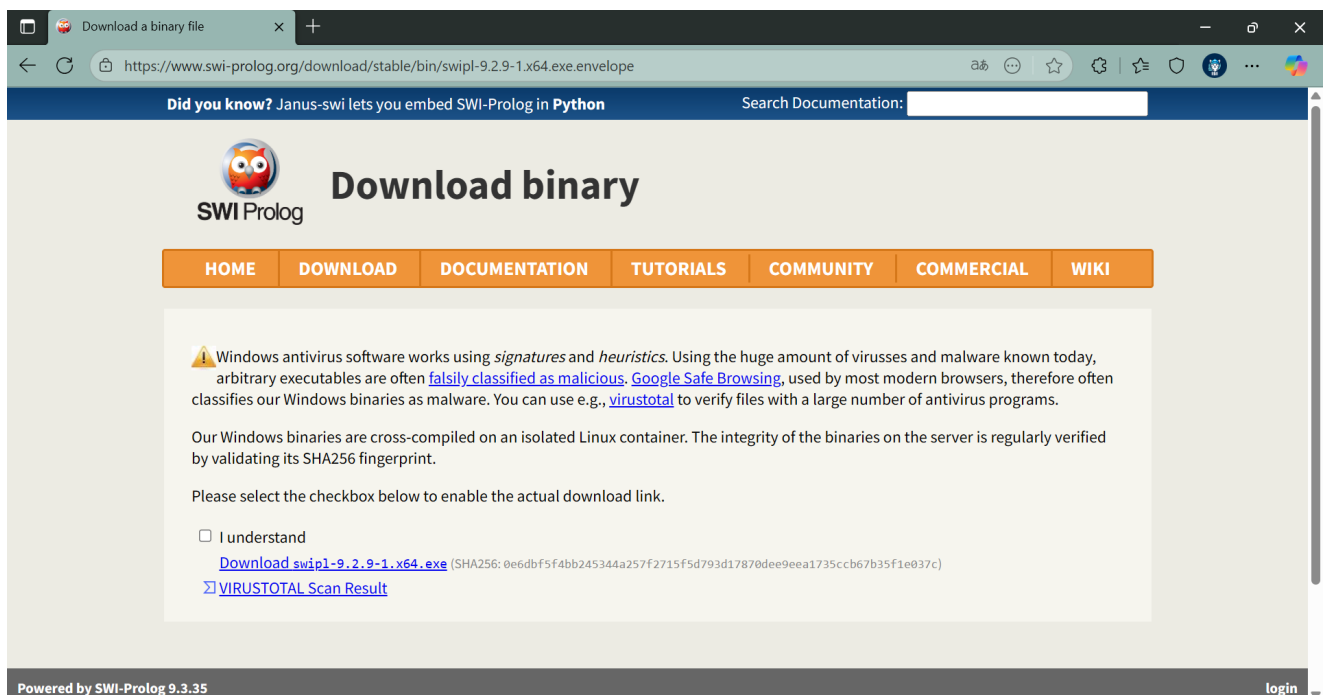


The screenshot shows the SWI-Prolog download page. At the top, there's a navigation bar with links: HOME, DOWNLOAD, DOCUMENTATION, TUTORIALS, COMMUNITY, COMMERCIAL, and WIKI. Below the navigation bar, there are several informational paragraphs with icons: Linux versions available as packages, Android binaries for Termux, Windows release notes, and a link to the ChangeLog. The main content is a table titled "Binaries" with three rows. Each row contains an icon, the file size, and a description of the binary along with its SHA256 fingerprint.

Binaries		
	14,331,585 bytes	<a href="#">SWI-Prolog 9.2.9-1 for Microsoft Windows (64 bit)</a> Self-installing executable for Microsoft Windows 64-bit editions. <b>SHA256:</b> 0e6dbf5f4bb245344a257f2715f5d793d17870dee9eea1735ccb67b35f1e037c
	14,020,721 bytes	<a href="#">SWI-Prolog 9.2.9-1 for Microsoft Windows (32 bit)</a> Self-installing executable for Microsoft Windows 32-bit editions. Version 9.3 is that last version of SWI-Prolog that is also released for 32-bit. Note that this version lacks the Janus interface to Python. <b>SHA256:</b> 1c9a87f2fd3ecc5311226b72a9b03989e500250ff469d7418f31706ce16b2de7
	39,839,539 bytes	<a href="#">SWI-Prolog 9.2.9-1 for MacOSX 10.14 (Mojave) and later on x86_64 and arm64</a> Mac OS X disk image with <a href="#">relocatable application bundle</a> . Needs <a href="#">xquartz</a> (X11) installed for running the <a href="#">development tools</a> . The bundle also provides the commandline tools in the <code>Contents/MacOS</code> directory. Users of older MacOS versions are adviced to use Macports, Homebrew or install from source. This bundle contains universal (fat) binaries that run natively on Intel (x86_64) and Apple Silicon (M1-3, arm64). <b>SHA256:</b> 90531c03e6c1182a0d5be45b865db6e399933e84b284dccc20ab931240420e37

Ya que selecciones tu sistema operativo, se te abrirá una pestaña.

- Marca la casilla **I understand**.



The screenshot shows the "Download binary" page for SWI-Prolog. It features a navigation bar with links: HOME, DOWNLOAD, DOCUMENTATION, TUTORIALS, COMMUNITY, COMMERCIAL, and WIKI. Below the navigation bar, there's a warning about Windows antivirus software and a note about the integrity of the binaries. A checkbox labeled "I understand" is present, followed by a link to download the binary and a link to the VirusTotal scan result. The footer shows "Powered by SWI-Prolog 9.3.35" and a "login" link.

**Download binary**

Windows antivirus software works using *signatures* and *heuristics*. Using the huge amount of viruses and malware known today, arbitrary executables are often [falsily classified as malicious](#). [Google Safe Browsing](#), used by most modern browsers, therefore often classifies our Windows binaries as malware. You can use e.g., [virustotal](#) to verify files with a large number of antivirus programs.

Our Windows binaries are cross-compiled on an isolated Linux container. The integrity of the binaries on the server is regularly verified by validating its SHA256 fingerprint.

Please select the checkbox below to enable the actual download link.

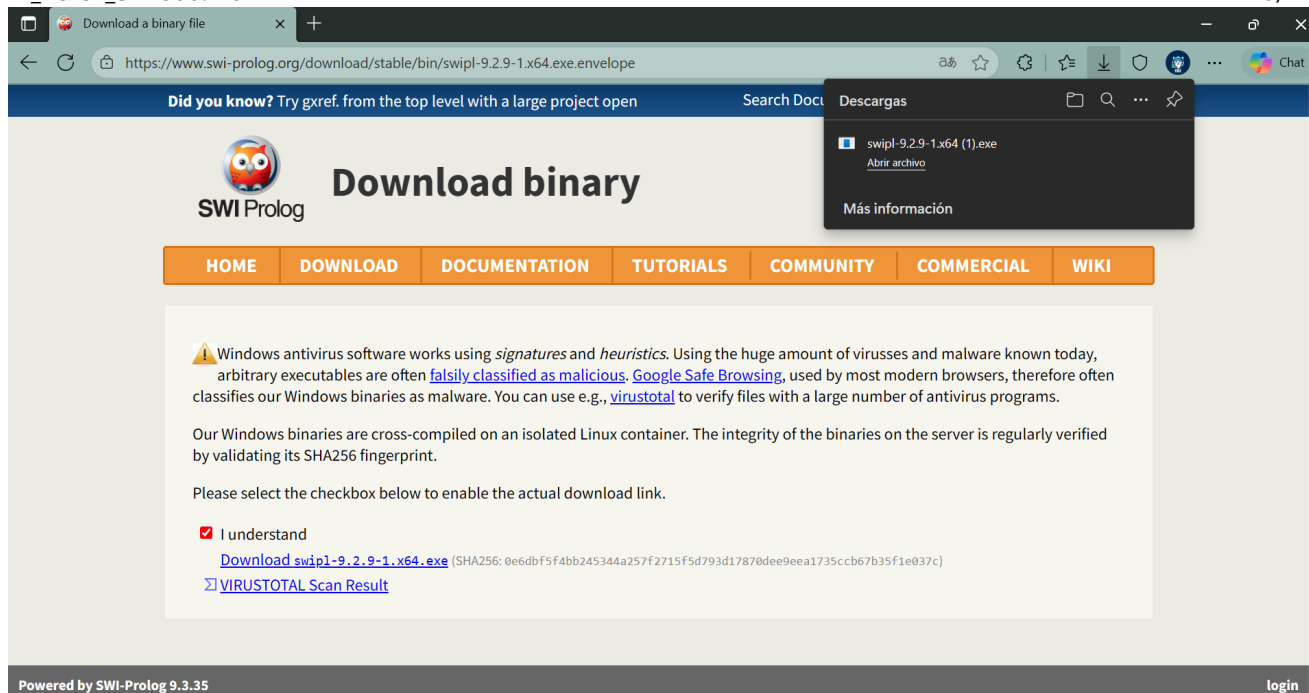
☐ I understand

[Download swipl-9.2.9-1.x64.exe](#) (SHA256: 0e6dbf5f4bb245344a257f2715f5d793d17870dee9eea1735ccb67b35f1e037c)

[VIRUSTOTAL Scan Result](#)

Powered by SWI-Prolog 9.3.35 [login](#)

Da clic en **Download swipl-9...**



## Conceptos Básicos

El paradigma lógico se diferencia fundamentalmente de otros paradigmas (como el funcional) en su estructura. Mientras que la programación funcional se centra en procedimientos y retornos, la programación lógica se basa en una "Base de Conocimiento" y una "Máquina" que procesa "Preguntas" para generar "Respuestas".

En Prolog, los elementos fundamentales son:

1. **Hechos (Facts):** Son afirmaciones incondicionales sobre objetos o relaciones.
  - **Sintaxis:** `relación(objeto1, objeto2)`. Los nombres de propiedades y objetos deben comenzar con minúscula y terminar con un punto.
  - *Ejemplos:* `cat(tom)`. (Tom es un gato) o `loves_to_eat(jorge, pasta)`.
2. **Reglas (Rules):** Definen relaciones condicionales (si-entonces).
  - **Sintaxis:** `cabeza :- cuerpo`. Se lee "La cabeza es verdad si el cuerpo es verdad".
  - *Ejemplo:* `happy(lili) :- dances(lili)`. (Lili es feliz si baila).
3. **Consultas (Queries):** Son preguntas que se hacen a la base de conocimientos para verificar la veracidad de una relación, como `¿Es Tom un gato?`.

## Programación Avanzada en Prolog

### Recursión

La recursión es vital en Prolog para recorrer estructuras o definir relaciones profundas, como la de "predecesor" (ancestro).

- Caso base: Un padre es un predecesor directo.
- Caso recursivo: Un predecesor es el padre de alguien que a su vez es predecesor del objetivo.

### Objetos de Datos

Prolog maneja varios tipos de objetos :

- Átomos y Constantes: Cadenas que empiezan con minúscula o entre comillas (`tom`, `'Hello World'`).
- Variables: Siempre comienzan con letra mayúscula (`X`, `Y`, `Resultado`).
- Números: Enteros y reales (`100`, `2000.45`).

## Operadores y Aritmética

Se utilizan operadores para comparaciones (`>`, `<`, `==` para igualdad numérica) y operaciones aritméticas estándar (`+`, `-`, `*`, `/`, `mod`) usando la palabra clave `is` para la asignación.

- Ejemplo: `X is 100 + 200..`

## Estructuras de Control y Listas

- **Bucles:** Se implementan mediante recursión y condiciones de parada, como contar hasta 10 o un rango `between(L, H, Y)`.
- **Listas:** Se representan con corchetes `[a, b, c]`. Se manipulan dividiéndolas en Cabeza (Head) y Cola (Tail) usando la sintaxis `[Head | Tail]`.
- **Operaciones de Listas:**
  - *Membresía:* `list_member(X, [X|_])..`
  - *Concatenación:* Unir dos listas.
  - *Eliminación y Permutación:* Manipulación de elementos dentro de la estructura .

## Predicados Integrados (Built-in)

El lenguaje incluye predicados útiles para verificar tipos de datos o realizar funciones matemáticas:

- `var(X)`: Verifica si X es una variable no instanciada.
- `atom(X)`: Verifica si es un átomo.
- `random(L,H,X)`: Genera un número aleatorio.

## Aplicaciones con Prolog

1. **Backtracking:** El mecanismo por el cual Prolog busca soluciones alternativas cuando falla una rama de ejecución.
2. **Estructuras de Datos Complejas:** Implementación de árboles y listas enlazadas.
3. **Resolución de Problemas Lógicos:**
  - *Torres de Hanoi:* Algoritmo recursivo clásico.
  - *Circuitos Resistivos:* Cálculo de valores en circuitos.
  - *El mono y el plátano:* Un problema clásico de planificación y estados en inteligencia artificial.

## Referencias

José Carlos Gallegos Mariscal, M. (2025-2). Unidad V El paradigma lógico.  
[https://drive.google.com/file/d/1hgdy11WRojvF\\_1wbcxWGkNH5QgwZiSO/view](https://drive.google.com/file/d/1hgdy11WRojvF_1wbcxWGkNH5QgwZiSO/view)