



## Ayudantía 1

# ASP y Clingo + Instalación Clingo

Por Jerónimo Infante  
y Juan José Alonso

16 de agosto de 2024



# Contenidos

## 1. Instalación Clingo

- a. macOS
- b. Linux
- c. Windows

## 2. ASP y Clingo

- a. ¿Qué es ASP y para qué se usa?
- b. Predicados
- c. Átomos / Proposiciones
- d. Modelo
- e. Reglas
- f. Anexo



# Instalación de Clingo



# Instalación de Clingo (macOS)

```
$ /bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/i
nsta11/HEAD/install.sh)"
(...)
$ xcode -select --install
(...)
$ brew install clingo
(...)
```

1. Instalar el administrador de paquetes *homebrew* (<https://brew.sh/>)
2. Instalar las herramientas de línea de comandos de *Xcode*
3. Instalar *clingo* mediante *homebrew*



# Instalación de Clingo (Linux)

```
$ sudo apt-get update  
(...)  
$ sudo apt-get install gringo  
(...)
```

1. Actualizar la lista de paquetes disponibles
2. Instalar *gringo* (que instalará a *clingo* como dependencia)

# Instalación de Clingo (Windows)



## 2.1.3. Instalación en Windows

La instalación en Windows es menos directa que la de otros sistemas operativos, deberemos añadir manualmente el programa al PATH de nuestro equipo.

1. Descarga clingo 5.4.0 desde [su página de descargas en GitHub](#) [10] o [directamente desde este link](#).
2. Extrae los componentes del archivo .zip, asegúrate de hacerlo en alguna ubicación permanente ya que los archivos deberán permanecer en ella desde ahora en adelante. Luego, copia la dirección en cuál los contenidos fueron extraídos.

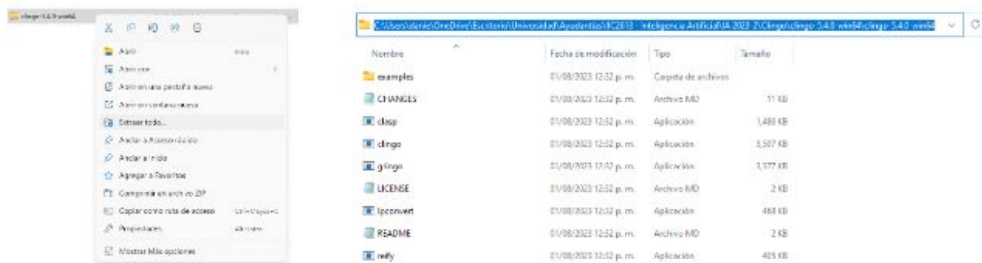


Figura 2.1: Se extraen los contenidos del archivo .zip y se copia el directorio.

# Instalación de Clingo (Windows)



3. En el buscador de Windows, buscaremos 'Editar las variables de entorno del sistema', abriendo el menú del panel de control y seleccionando el botón en la esquina inferior derecha de 'Variables de entorno...', en este nuevo menú haremos click en PATH y luego en el botón 'Editar...'

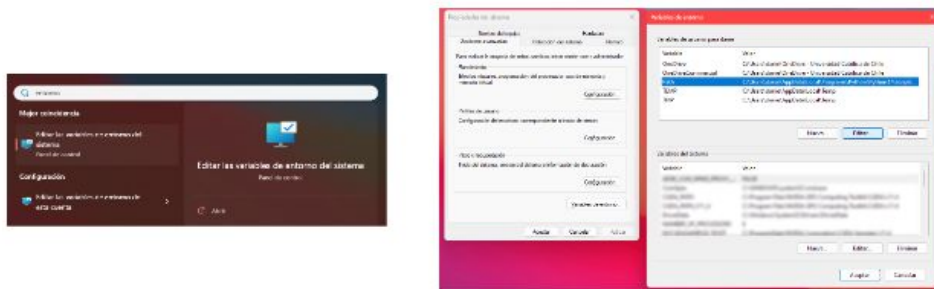


Figura 2.2: Se abre el menú de variables de entorno y se editan las variables en el PATH.

# Instalación de Clingo (Windows)



4. Dentro de la pestaña que acaba de abrirse, haremos click en 'Nuevo' para crear un nuevo directorio en el PATH y pegaremos el directorio que copiamos anteriormente (en el que extraímos el .zip). Hacemos click en 'Ok' y ahora podremos utilizar clingo correctamente.

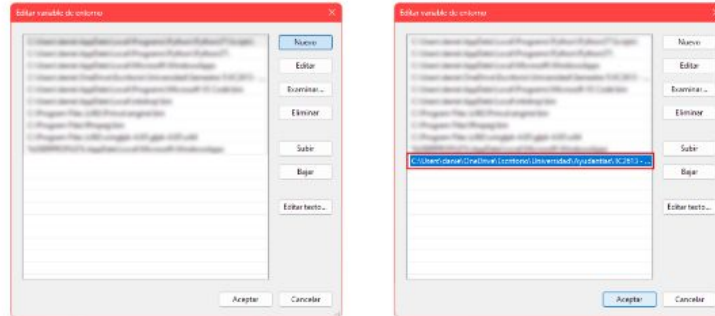


Figura 2.3: Añadimos el directorio de clingo a nuestro PATH.





# Instalación de Clingo

```
$ clingo
```

```
clingo version 5.7.1
```

```
Reading from stdin
```



# Instalación de Clingo

```
$ clingo
```

```
clingo version 5.7.1
```

```
Reading from stdin
```



# ASP y Clingo



# ¿Qué es ASP y para qué se usa?

**Answer Set Programming**

**=**

**Programación de Conjuntos de Respuestas**

- Paradigma de programación lógica
- Busca conjuntos de elementos que cumplan con ciertas reglas o propiedades



# ¿Qué es ASP y para qué se usa?

Ejemplo clásico de lógica proposicional:

*"Todos los hombres son mortales"*

y

*"Sócrates es hombre"*



*"Sócrates es mortal"*



# ¿Qué es ASP y para qué se usa?

## Programación declarativa

- Describe hechos como:
  - *"El sol existe"*
  - *"El sol emite radiación"*
- Lenguajes como *clingo*

```
existe(sol).
```

## Programación imperativa

- Da ordenes como:
  - *"Suma 6 + 17"*
  - *"Imprime 'Hello world!'"*
- Lenguajes como *Python*

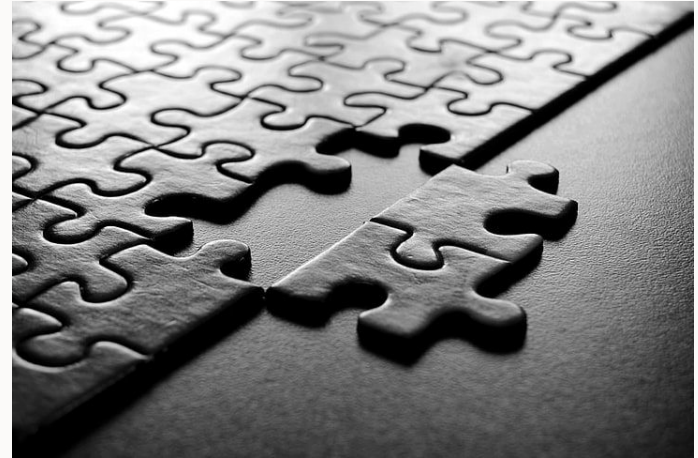
```
print("Hello world!")
```



# ¿Qué es ASP y para qué se usa?

Se usa para resolver:

- Problemas de búsqueda
- Puzzles lógicos
- Problemas que requieran usar el **sistema 2** → pensamiento normativo





# ¿Qué es ASP y para qué se usa?

## Clingo

- Es un lenguaje que combina **ASP** con solucionadores de satisfacibilidad **SAT**
- Es el lenguaje que usaremos para escribir programas lógicos
- Sus archivos tienen extensión **.lp** y, para ejecutarlos, se debe escribir en consola

```
clingo {nombre_archivo}.lp
```





# Predicados

- Constantes que representan una propiedad, relación, o característica con sus términos
- **Siempre** comienzan con **minúscula**

```
existe(sol).      % Tiene una constante simbólica
existe(1).        % Tiene una constante numérica
existe(X).        % Tiene una variable
```

**Ojo:** las variables solo existen dentro de los predicados, y siempre comienzan con **mayúscula**



# Predicados

## Aridad

- Corresponde al **número de términos** que reciben

```
p.                % Predicado nulario (aridad 0)
ayudante(juanjo). % Predicado unario  (aridad 1)
amigo(shrek, burro). % Predicado binario (aridad 2)
```



# Átomos / Proposiciones

- Definen propiedades o reglas que pueden ser **verdaderas** o **falsas**
- Un mismo predicado puede definir múltiples proposiciones, si se definen con la misma palabra pero distinta aridad

`p.`

`p(q).`

`aprende(estudiante).`

`aprende(estudiante, profesor).`



# Modelo

- Es la **solución** del programa lógico
- Es un **conjunto minimal** de átomos que satisfacen las condiciones lógicas
- Pueden existir **varios**, así como **ninguno**

Ejemplo de output  
de *Clingo*

```
Solving...  
Answer: 1  
p r q  
SATISFIABLE
```

```
Models      : 1
```

El modelo obtenido  
**{p, r, q}**



# Modelo

## Minimalidad

- Solo son modelos aquellos conjuntos con la **mínima cantidad** posible de átomos
- De lo contrario, podrían existir infinitos modelos
- Para el ejemplo anterior, si  $\{p, r, q\}$  es un modelo,  $\{p, r, q, s\}$  no puede serlo



# Reglas

*Head*  $\leftarrow$  *Body*

- Si *Body* es verdadero, algo en *Head* también debe serlo
- Tanto *Head* como *Body* son conjuntos de átomos o proposiciones
- Se pueden construir hechos a partir de reglas que carezcan de *Body*

```
llueve.  
mojado(niño) :- llueve.  
enojado(niño) :- mojado(niño).           % lamentable :(
```

¿Cuál o cuáles son los modelos de este programa?



# Reglas

*Head*  $\leftarrow$  *Body*

- Si *Body* es verdadero, algo en *Head* también debe serlo
- Tanto *Head* como *Body* son conjuntos de átomos o proposiciones
- Se pueden construir hechos a partir de reglas que carezcan de *Body*

```
llueve.  
mojado(niño) :- llueve.  
enojado(niño) :- mojado(niño).           % lamentable :(
```

El modelo es **{llueve, mojado(niño), enojado(niño)}**



# Reglas

## Body con varios átomos

- Generan una **conjunción** de proposiciones; es decir, se debe cumplir todo en *Body* para que la regla se exija

```
a.          % a se encuentra en el modelo
b.          % b se encuentra en el modelo
c :- a, b.   % c está sólo si a y b lo están
d :- a, m.   % d está sólo si a y m lo están
```

¿Cuál o cuáles son los modelos de este programa?





# Reglas

## Body con varios átomos

- Generan una **conjunción** de proposiciones; es decir, se debe cumplir todo en *Body* para que la regla se exija

```
a.          % a se encuentra en el modelo
b.          % b se encuentra en el modelo
c :- a, b.   % c está sólo si a y b lo están
d :- a, m.   % d está sólo si a y m lo están
```

El modelo es **{a, b, c}**




# Reglas

## Head con varios átomos

- Generan una **disyunción** de proposiciones; es decir, cuando se cumple el *Body*, se cumple sólo uno de los átomos del *Head*
- A excepción de que se fuerce la presencia de más átomos

```
p.  
q, r, k :- p.
```

¿Por qué sucede esto?

 Hint: minimalidad



# Reglas

## Predicados con variables

- Permiten definir múltiples proposiciones de manera simultánea.

```
pajaro(carpintero).  
pajaro(martin_pescador).  
pajaro(condor).  
vuela(carpintero).  
vuela(martin_pescador).  
vuela(condor).
```

Esto...

```
pajaro(carpintero).  
pajaro(martin_pescador).  
pajaro(condor).  
vuela(Z) :- pajaro(Z).
```

...es equivalente a esto



# Ejercicios

## Ejercicio 2.3.1: Reglas básicas

```
p :- p, q. % si p y q se encuentran en el modelo, p también  
r :- s, t. % si s y t se encuentran en el modelo, r también  
q :- s.    % si s se encuentra en el modelo, q también  
p.         % p se encuentra en el modelo
```

¿Qué modelo es la única solución a este programa??

- a) {p}
- b) {p, q}
- c) Vacío
- d) El problema es insatisfacible (no existe un modelo que lo solucione)

*Ejercicio obtenido de 'Intro a la Inteligencia Artificial' por Daniel Florea*



# Ejercicios

## Ejercicio 2.3.1: Reglas básicas

```
p :- p, q. % si p y q se encuentran en el modelo, p también  
r :- s, t. % si s y t se encuentran en el modelo, r también  
q :- s.    % si s se encuentra en el modelo, q también  
p.         % p se encuentra en el modelo
```

¿Qué modelo es la única solución a este programa??

- a) {p}
- b) {p, q}
- c) Vacío
- d) El problema es insatisfacible (no existe un modelo que lo solucione)

*Ejercicio obtenido de 'Intro a la Inteligencia Artificial' por Daniel Florea*



# Ejercicios

```
q.           % q se encuentra en el modelo
p :- q, r.    % si q y r se encuentran en el modelo, p también
r :- p.       % si p se encuentra en el modelo, r también
```

¿Qué modelo genera este programa?

*Ejercicio inspirado de 'Intro a la Inteligencia Artificial' por Daniel Florea*



# Ejercicios

```
q.           % q se encuentra en el modelo
p :- q, r.   % si q y r se encuentran en el modelo, p también
r :- p.      % si p se encuentra en el modelo, r también
```

El modelo que genera es **{q}**

*Ejercicio inspirado de 'Intro a la Inteligencia Artificial' por Daniel Florea*



# Anexo

## Uso de *statements*

#show

Muestra en las respuestas solo los átomos que nos interesan.

#show predicado/aridad.

```
triste(niño) :- mojado(niño).  
mojado(niño) :- llueve.  
llueve.  
#show triste/1.
```

#const

Permite reemplazar términos de constantes.

Se puede hacer directo por consola también con -c constante=valor

```
grande(c0)  
#const c0 = 64.
```





# Anexo

- Para que el programa muestre n modelos al ejecutar, se debe escribir en consola el número **N** al final del comando de ejecución.
- Si se quieren ver todos los modelos posibles, se debe escribir el número 0.

```
clingo {nombre_archivo}.lp N
```



# Anexo

## ¿Por qué una regla sin Body es verdadera?

Se puede considerar que el Body vacío es igual a Falso, luego en una implicancia lógica, la tabla de verdad es la siguiente:

$p$	$q$	$(p \Rightarrow q)$
V	V	V
V	F	F
F	V	V
F	F	V

La implicancia también se puede escribir como una disyunción:

$$\neg p \vee q$$

$$\neg v \vee v = \text{falso o verdadero} = \text{verdadero}$$

$$\neg v \vee f = \text{falso o falso} = \text{falso}$$

$$\neg f \vee v = \text{verdadero o verdadero} = \text{verdadero}$$

$$\neg f \vee f = \text{verdadero o falso} = \text{verdadero}$$



# Anexo

Libros/apuntes para más información:

- <https://github.com/dfloreaa/Apuntes-IIC2613/blob/main/Intro%20a%20la%20Inteligencia%20Artificial%20-%20Daniel%20Floreaga.pdf> (en proceso de escritura)
- <https://www.cs.utexas.edu/users/vl/teaching/378/ASP.pdf>
- [http://wp.doc.ic.ac.uk/arusso/wp-content/uploads/sites/47/2015/01/clingo\\_guide.pdf](http://wp.doc.ic.ac.uk/arusso/wp-content/uploads/sites/47/2015/01/clingo_guide.pdf)



## Ayudantía 1

# ASP y Clingo + Instalación Clingo

Por Jerónimo Infante  
y Juan José Alonso

16 de agosto de 2024