



Manual de usuario Wazelog

Índice

I.Introducción.....	3
1. Objetivo.....	3
2. Requerimientos.....	3
3. Versión.....	3
II.Desarrollo del manual de usuario.....	4
Abrir la aplicación con SWI-Prolog.....	4
Abrir la aplicación Visual studio Code/ Terminal.....	7
Uso del programa.....	12
III.Limitaciones.....	13
1.Limitación en la frase del cliente.....	13
2. Limitación del grafo.....	14
3. Tipo de oraciones y palabras registradas en el programa.....	15
Oraciones.....	15
Palabras.....	16
4. Ejemplos de frases aceptadas por el programa.....	20

I. Introducción

1. Objetivo

Establecer los pasos necesarios para abrir el programa WazeLog y los pasos necesarios que se necesitan para utilizar este programa, además de mostrar ejemplos de oraciones que son posibles a utilizar en este programa debido al BNF definido por los desarrolladores, además de mostrar las limitaciones en las frases que se envían a prolog como puede ser palabras separadas en caso de nombre de ciudades.

2. Requerimientos

- Equipo Pentium II o superior
- Mínimo 1GB de RAM
- Sistema Operativo Windows 10 o superior
- SWI-Prolog 9.0.4 o superior

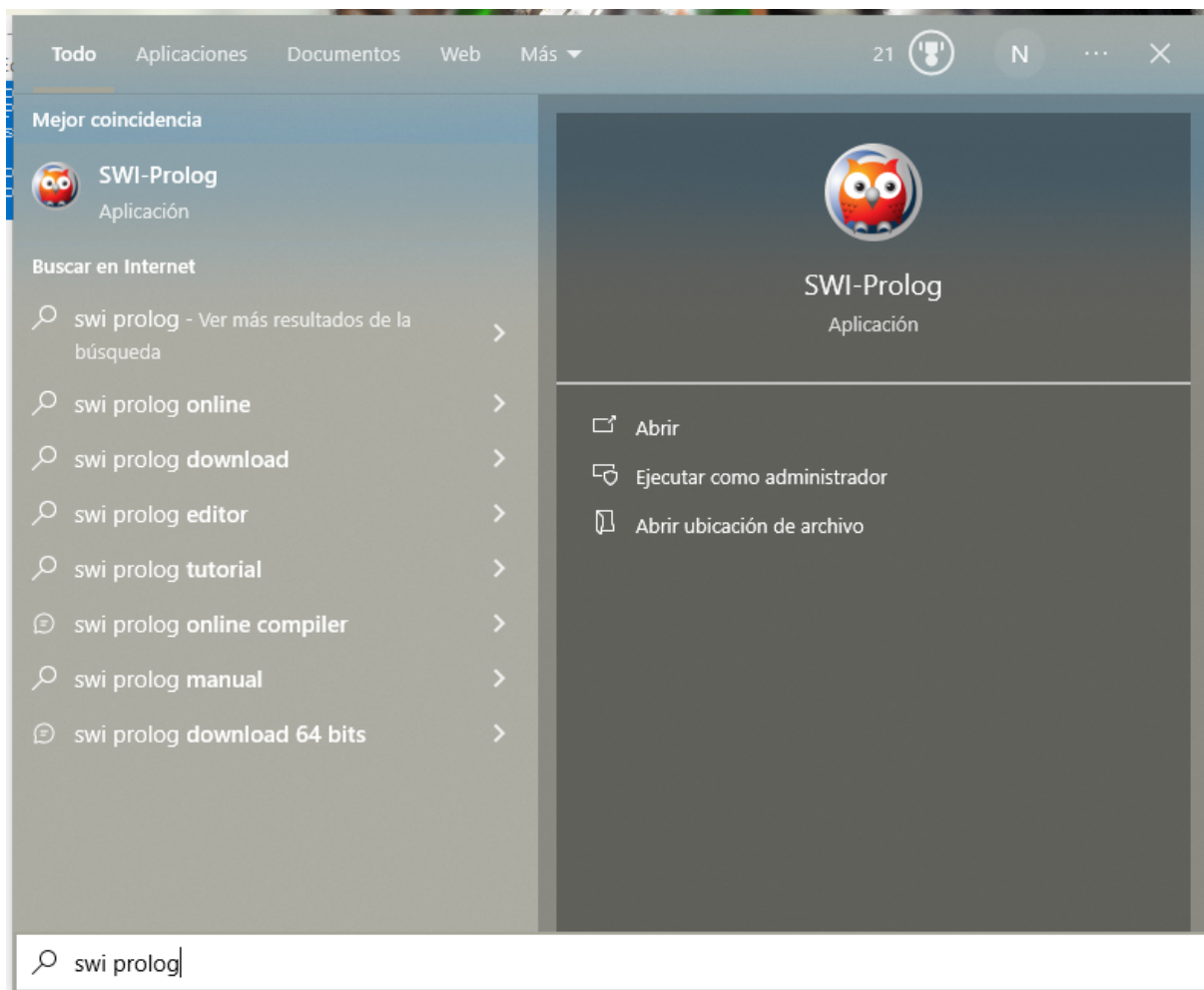
3. Versión

Esta es la primera versión y única hasta el momento de Wazelog, donde se limita a un simple grafo el cual se encuentra en el apartado de limitaciones de este manual de usuario

II.Desarrollo del manual de usuario

Abrir la aplicación con SWI-Prolog

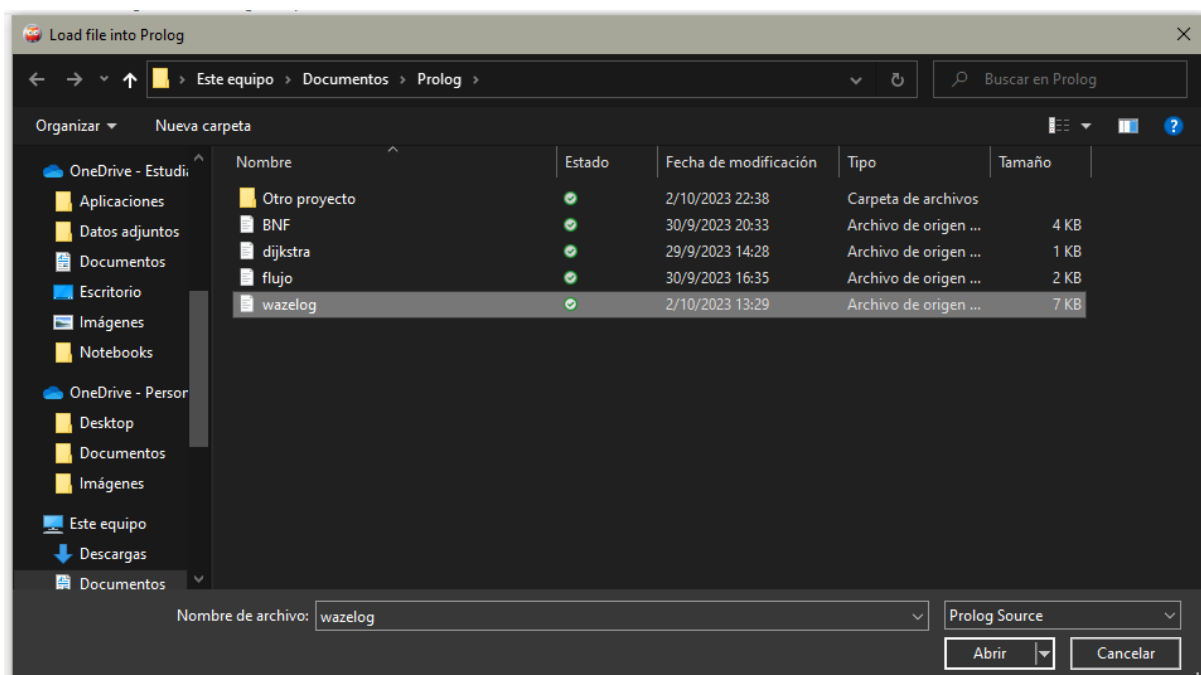
1. Antes que todo debemos tener instalado Prolog, en este caso también el IDLE de SWI-Prolog que podemos encontrar en su página oficial de Prolog. Lo buscamos en nuestro ordenador



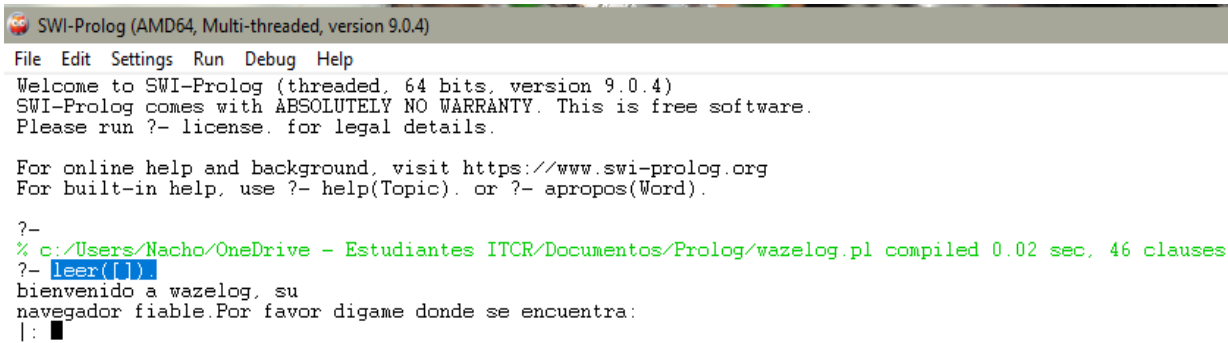
- Al abrirse esta aplicación podemos buscar la opción consult en la parte superior izquierda donde dice File



- Luego tenemos que buscar la dirección donde descargamos nuestros archivos de wazelog, sabiendo que tenemos todos los archivos en la misma carpeta tenemos que elegir wazelog donde estará la parte principal del programa donde iniciaremos la aplicación



4. Una vez elegido el archivo principal que en este caso es wazelog nos compilara el programa lo cual solamente tenemos que poner el siguiente comando en la consola de SWI-Prolog: leer([]).



```
SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

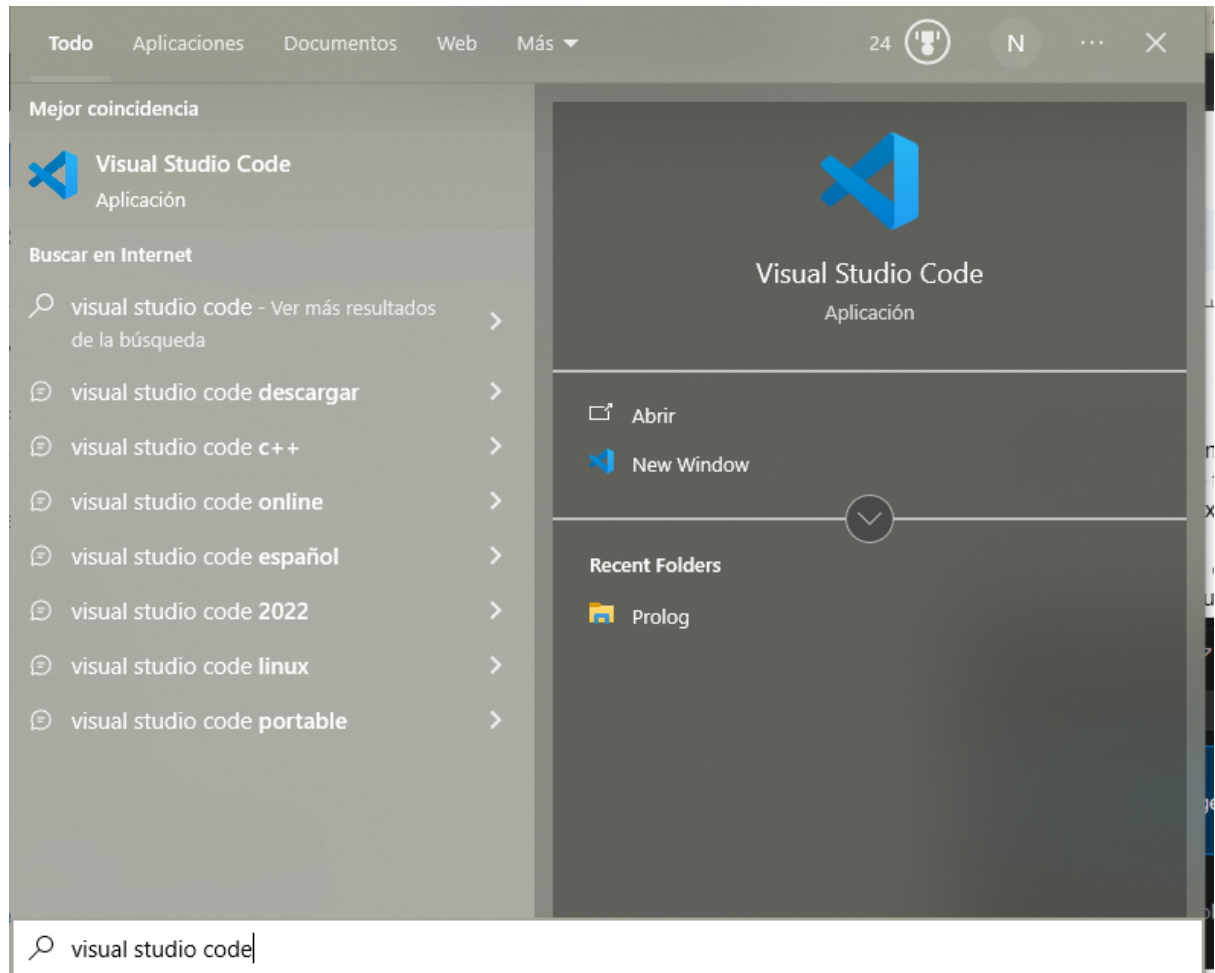
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
% c:/Users/Nacho/OneDrive - Estudiantes ITCR/Documentos/Prolog/wazelog.pl compiled 0.02 sec, 46 clauses
?- leer([]).
bienvenido a wazelog, su
navegador fiable.Por favor digame donde se encuentra:
|: █
```

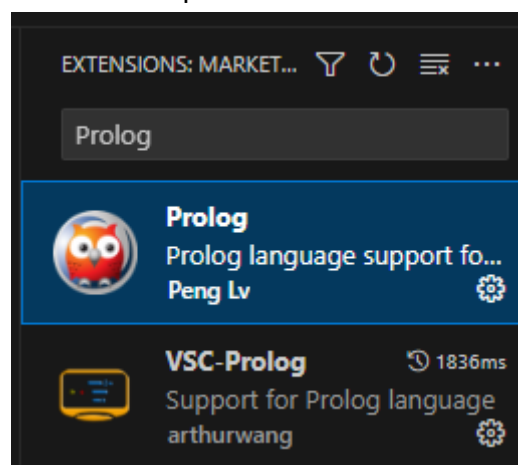
Así iniciamos nuestro programa WazeLog correctamente!

Abrir la aplicación Visual studio Code/ Terminal

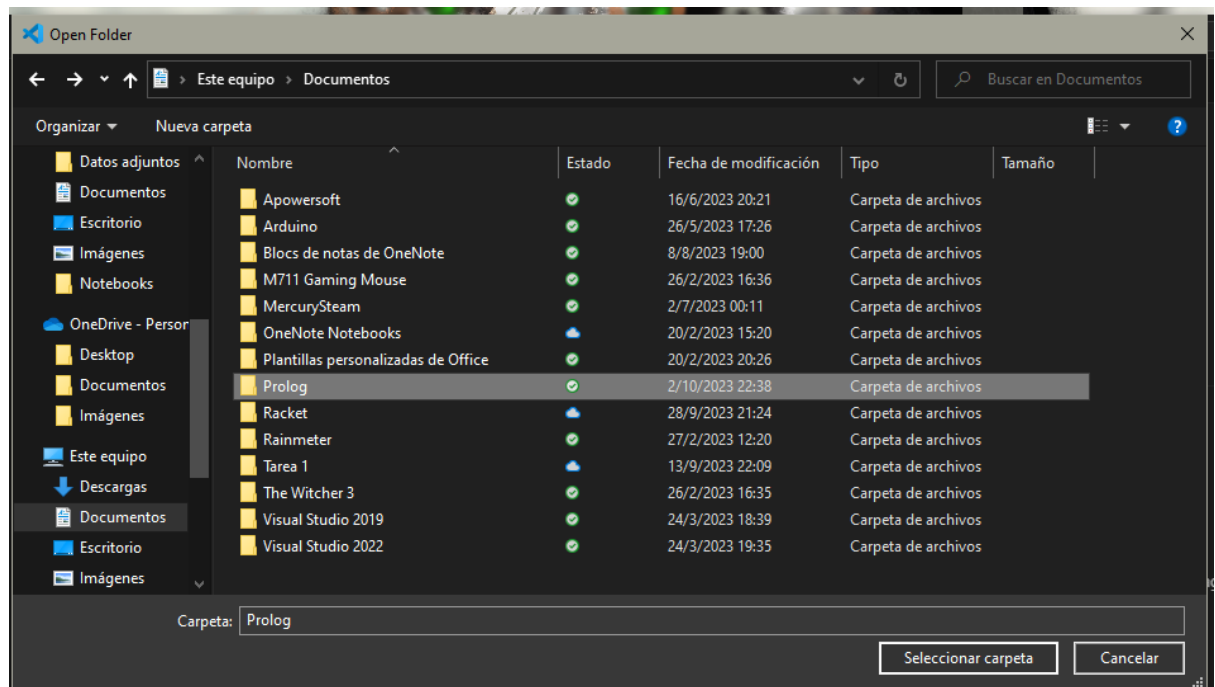
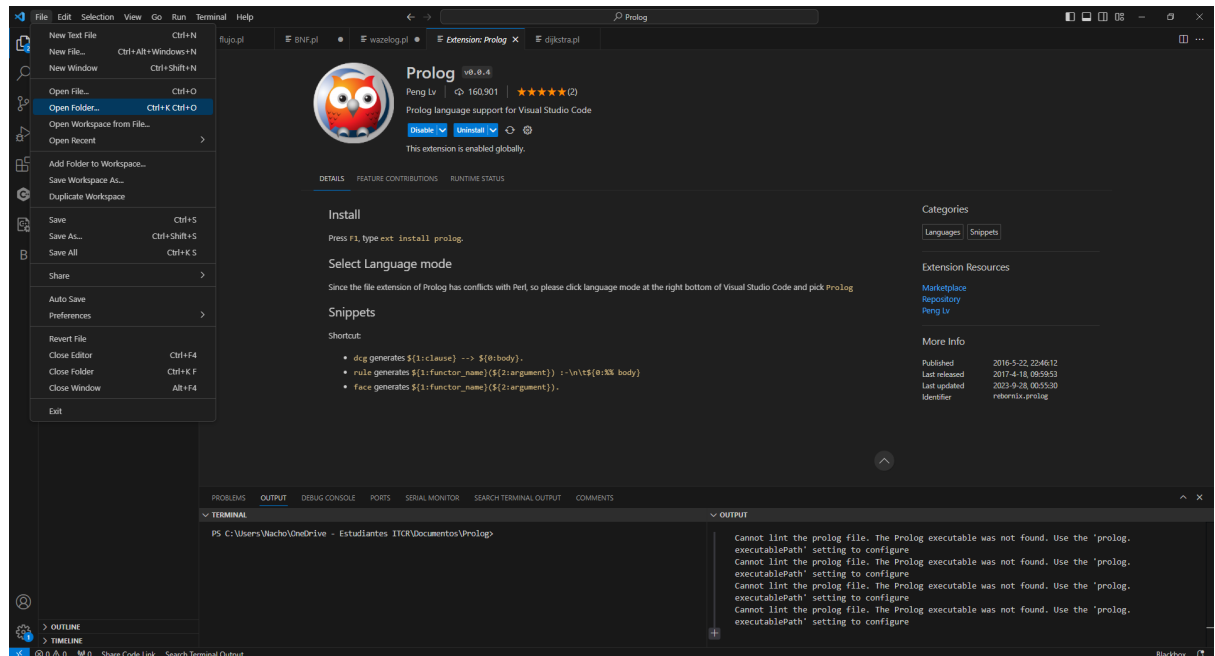
1. Para hacerlo de esta manera necesitamos tener Visual Studio Code en su versión más reciente, además de tener prolog instalado en nuestra computadora, además tenemos que instalar la extensión de este en visual studio code



2. Necesitamos instalar la extensiones de Prolog, podemos usar cualquiera de las 2 siguientes extensiones que encontramos en visual studio code

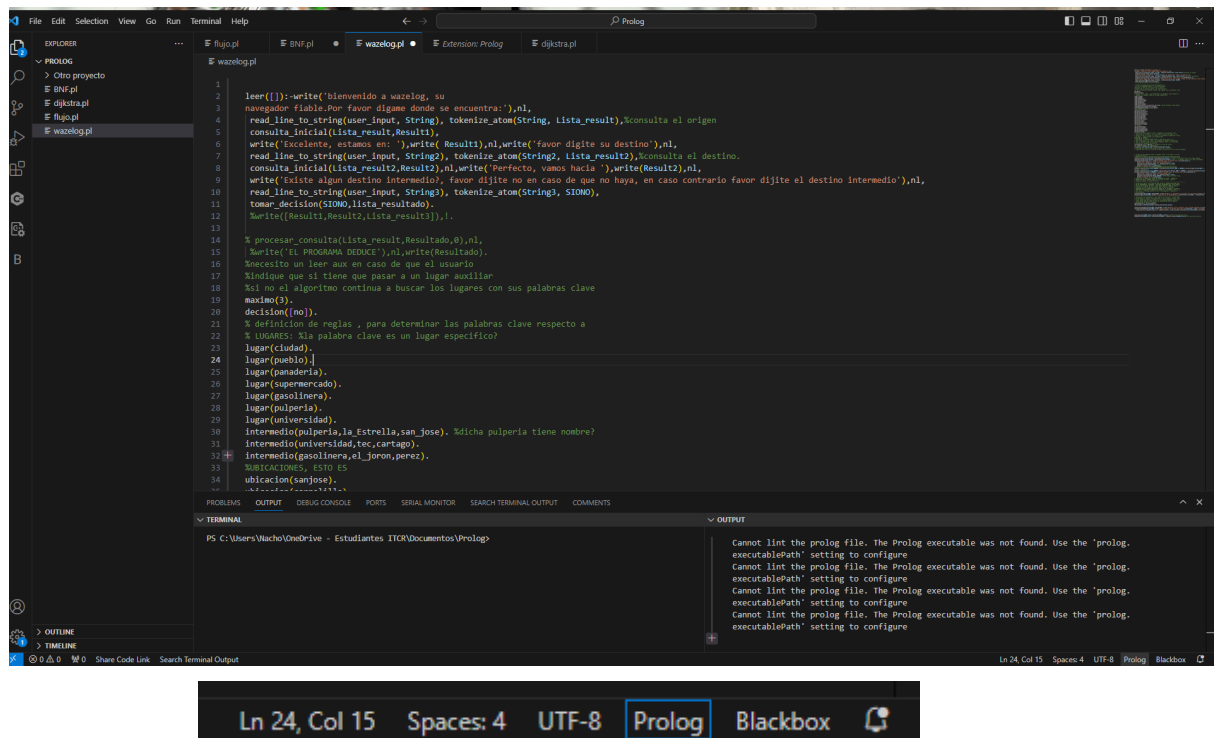


3. Luego necesitamos encontrar donde tenemos los archivos de nuestro programa de prolog, podemos hacerlo desde visual studio code de la siguiente manera.

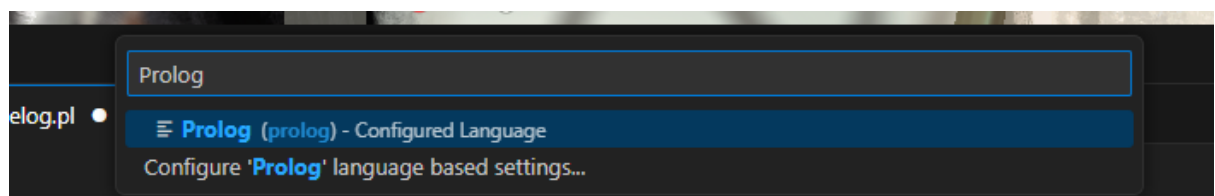


Necesitamos seleccionar la carpeta donde tenemos los archivos en este caso está en la carpeta llamada Prolog

- Luego de esto necesitamos seleccionar el archivo llamado wazelog.pl en la parte izquierda donde tendremos un panel con los diferentes archivos del programa wazelog, una vez que estamos en el archivo necesitamos poner el lenguaje de Prolog correctamente ya que visual studio code leerá como si fuera un archivo de Perl.

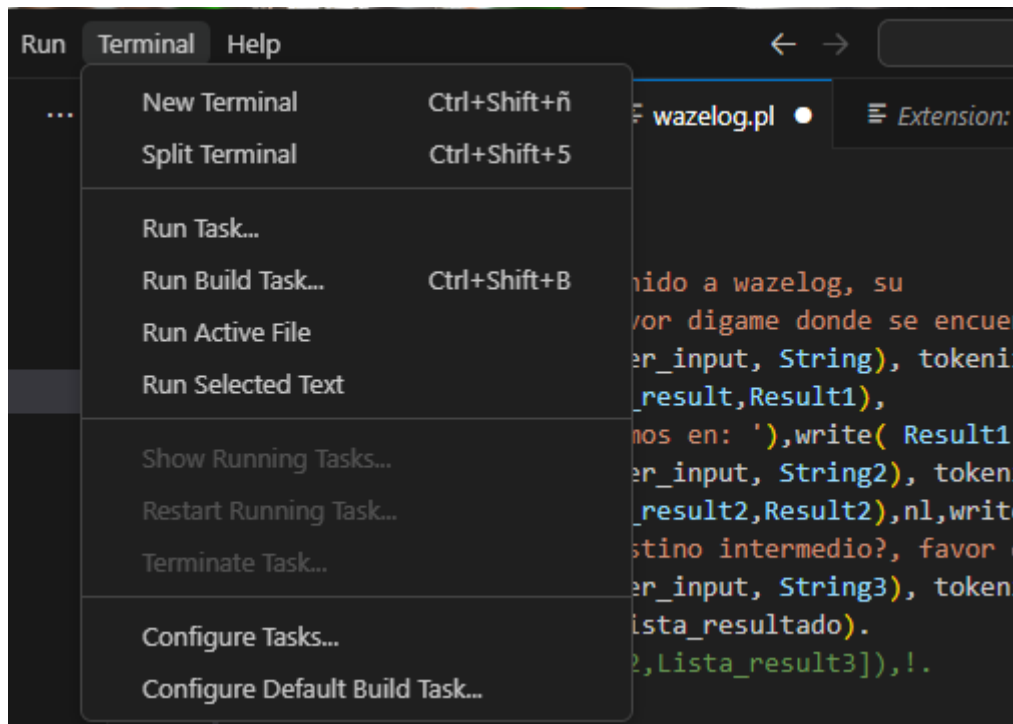


En la parte inferior derecha es donde debería decir Perl, en este caso dice Prolog por que ya estaba bien configurado anteriormente el visual studio code, una vez el mouse por encima le podemos dar clic y nos abrirá un buscador en la parte media superior de visual studio code, donde buscaremos Prolog y elegiremos la opción que nos aparece

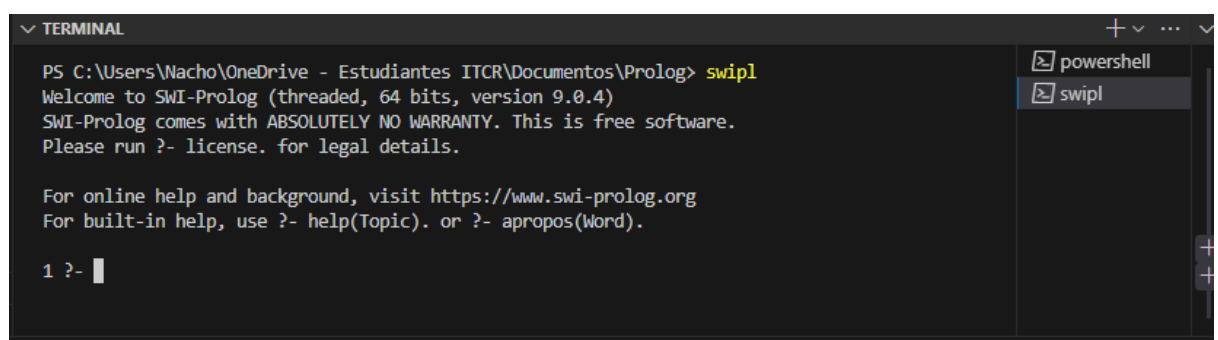


Con esto ya tendremos configurado Prolog para nuestro proyecto!

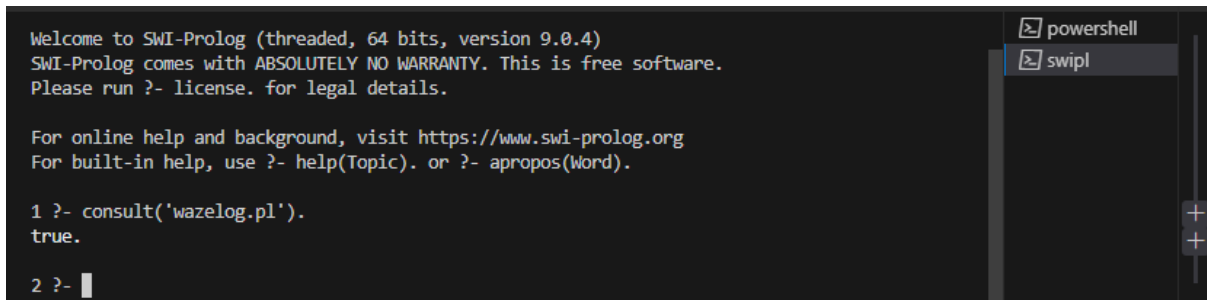
- Lo siguiente es abrir una nueva terminal para que esté en la dirección de la carpeta, esto podemos hacerlo con visual studio code, en la opción de terminal y en New terminal, o simplemente usando la combinación de teclas de Ctrl+Shift+ñ



- En este nuevo terminal que abrimos necesitamos escribir primero swipl que abrirá el compilador por decirlo así de SWI-Prolog



7. Como penúltimo paso para correr el programa wazelog necesitamos consultar el archivo de wazelog.pl que seria escribiendo el siguiente comando: `consult('wazelog.pl')`.



```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- consult('wazelog.pl').
true.

2 ?- 
```

Si aparece true. es que el programa se compiló sin ningún error crítico y de manera correcta!

8. Ahora simplemente necesitamos iniciar nuestro programa escribiendo el siguiente comando y usarlo!
el comando sería: `leer([])`.



```
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- consult('wazelog.pl').
true.

2 ?- leer([]).
bienvenido a wazelog, su
navegador fiable.Por favor digame donde se encuentra:
|: 
```

¡Ya tendríamos el programa Wazelog funcionando correctamente!

Uso del programa

Una vez que tenemos abierta nuestra aplicación de la manera que preferimos solo queda utilizarla contestando las preguntas que nos hace el chatbot wazelog para encontrar nuestra ruta más corta, es importante decir que tanto las oraciones como palabras que se pueden utilizar para responder las preguntas tiene ciertas limitaciones, las cuales podemos encontrar en este manual de usuario en el punto 3 de las limitaciones, ahí también podemos encontrar ejemplos que nos pueden ayudar además de si queremos crear nuestras propias oraciones podemos ver el orden y sintaxis que deben seguir además de las palabras que podemos utilizar

Es importante recordar también que este programa es la única versión y está creada a partir de un grafo, por cuales las rutas son únicamente con las ciudades del grafo que también podemos encontrar en el punto 2 de limitaciones, además de esto tenemos que tener en cuenta las limitaciones al usar nombre de ciudades que son separadas o llevan puntuaciones en alguna de sus letras ya que el programa no las puede reconocer bien o dar error por esto, como podemos apreciar también en el punto 1 de las limitaciones del programa

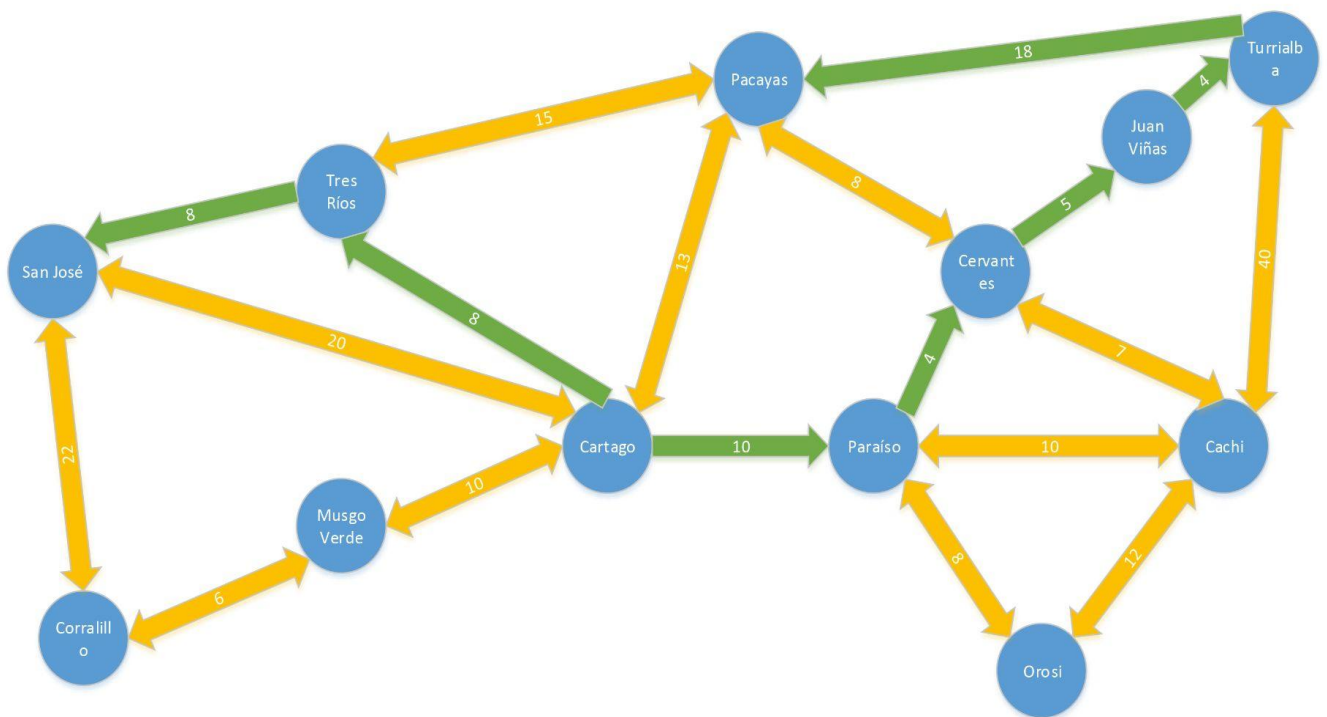
III.Limitaciones

1.Limitación en la frase del cliente

- Este programa tiene ciertas limitaciones en el momento que enviamos frases, por ejemplo en nombres de ciudades como San José o Tres Ríos, tienen que ser escritos de manera junta, sin mayúsculas y sin tildes, en este caso sería sanjose y/o tresrios.
- Siguiendo con el punto anterior todas las palabras ingresadas por el cliente tienen que ser en su totalidad en minúsculas, no pueden haber mayúsculas en estas, esto debido a que prolog no puede leer bien las mayúsculas y puede dar problema con el programa en el proceso de las diferentes acciones que hace.
- Además no pueden contener letras con tildes u otros tipo de puntuaciones en letras, ya que lo mismo que al tener mayúsculas podría dar errores de programa o que no lea bien los lugares por como lo lee Prolog.
- Las palabras que se pueden utilizar está limitado a una lista no muy extensa, debido a como está creado el programa con un BNF, por lo cual las oraciones utilizadas como se podía ver en el punto 2 del desarrollo del manual están limitadas a cierto tipo de oraciones y conformadas por ciertas palabras. se podrá encontrar las frases y tipo de oraciones en el 3 punto de las limitaciones

2. Limitación del grafo

El programa está limitado en las ciudades que uno puede pedir las rutas, en este caso se utilizó el siguiente grafo, por lo cuál el programa está limitado y funciona siguiendo las instrucciones de este grafo.



3. Tipo de oraciones y palabras registradas en el programa

Como se ha dicho anteriormente el programa solo recibe algún tipo de oraciones y algunas palabras para formar estas, esto debido a que el programa está hecho con un BNF creado desde 0. por lo cuál se tiene que registrar los tipos de oraciones que puede recibir, creando desde oraciones simples hasta oraciones más complejas. En esta parte se muestra los tipos de oraciones con algunos ejemplos de uso para la facilidad del usuario, también se mostrará el orden y sintaxis que tienen que tener las oraciones para que sean válidas para el programa, además de las palabras que están registradas en el programa por si quiere crear nuevas combinaciones

Oraciones

Las oraciones contienen sintagmas tanto verbales como nominales, primero que todo mostraremos las diferentes definiciones de estos sintagmas para entender cómo son conformadas las oraciones y el orden de estos ya que los sintagmas también son conformados en este caso por palabras más cortas y con cierto orden también

Sintagmas Verbales:

- verbo,verbo
- verbo,sintagma nominal
- verbo
- verbo,sintagma verbal
- pronombre,verbo
- verbo,conjunción
- verbo,adjetivo
- sustantivo,verbo
- verbo,sustantivo
- pronombre,sintagma verbal
- artículo indefinido,verbo

Sintagmas nominales:

- preposición,nodo
- nodo
- lugar
- artículo indefinido,lugar
- preposición, sintagma nominal
- verbo,nodo
- verbo,sintagma nominal

Ahora las oraciones que podemos hacer teniendo en cuenta como estan formados los sintagmas y también debemos tener en cuenta son las siguientes oraciones

Oraciones

- saludo
- afirmacion
- negacion
- despedida
- despedida,despedida
- nodo
- sintagma nominal
- lugar
- sintagma verbal,sintagma nominal
- saludo,sintagma verbal,sintagma nominal
- saludo,nombre,sintagma verbal,sintagma nominal

Palabras

Las palabras están predefinidas en el programa, como verbos, conectores, lugares,nodo que este último hace referencia a los nodos, a continuación podemos ver cada parte y las palabras que están definidas en cada una de esas partes y que se pueden utilizar para crear oraciones en el programa

Nodo:

- cartago
- corralillo
- sanjose
- musgoverde
- tresrios
- pacayas
- cervantes
- paraíso
- turrialba
- cachi
- orosi
- juanvinas

saludo:

- hola
- buenosdias
- buenas

despedida:

- hastaluego
- adios
- nosvemos

verbos:

- ir
- estoy
- quiero
- pasar
- necesito
- voy
- encuentro
- tengo

Más verbos:

- dirijo
- pasar
- hacer
- viaje
- conocer
- interesa
- visitar
- gustaria

nombres:

- wazelog

lugares:

- supermercado
- gasolinera
- farmacia
- cine

afirmacion:

- si
- porsupuesto

negacion:

- no
- ninguno

conjuncion:

- que

artículo indefinido:

- un
- una

sustantivos:

- destino
- ubicacion
- ganas
- sitio
- lugar

adjetivos:

- posicionado
- establecido
- localizado
- instalado

pronombre:

- me
- mi

preposiciones:

- en
- a
- hasta
- de
- por
- hacia

4. Ejemplos de frases aceptadas por el programa

- quiero ir a cartago
- necesito ir hasta cartago
- Estoy en cartago
- necesito pasar a supermercado
- necesito pasar a un supermercado
- necesito ir hacia cartago
- voy a cartago
- me encuentro en cartago
- tengo que ir a cartago
- me dirigo a cartago
- estoy posicionado en cartago
- mi destino es cartago
- mi ubicacion es cartago
- estoy ubicado en cartago

Otros sinónimos de frases aceptadas, bastante más complicadas

- me gustaria visitar cartago
- quiero hacer un viaje a cartago.
- tengo ganas de conocer cartago
- tengo ganas de conocer un cine
- tengo ganas de pasar a un cine

Es importante recordar que las frases tienen que ser totalmente en minúsculas, sin tildes, u otro tipo de acentuaciones además de no utilizar “ñ” tampoco.

cartago es como referencia a la ciudad o nodo en estos ejemplos, puede ser cambiado por otro de los tantos nodos o ciudades del grafo, igual que cine y/o

supermercado pueden ser cambiados por alguno de los otros tantos lugares registrados en el programa que pudimos ver anteriormente