Instala el entorno de Anaconda para ejecutar programas en Python y contesta a las preguntas marcadas en rojo:

1. Abre el programa llamado python_lists_simplicity.py de PIA_S01_Python_Lists_Simplicity.zip y contesta a las siguientes preguntas:

   1.1.¿Es Python un lenguaje que facilita el uso de listas?

   Teniendo previo conocimiento de otros lenguajes como Java y JavaScript en lo que se refiere al manejo de listas, en Python lo veo más simple. Pero debido a mi falta de experiencia con este lenguaje veo un poco de confusión a la hora de emplear las listas.

   1.2.¿Te parece Python un lenguaje que a simple vista es legible?

   De nuevo, comparado con los lenguajes mencionados, en Python las líneas de código las noto más legibles por su simplicidad. Por ejemplo, en la línea 11 que define un bucle *for*, que itera diez veces generando un número aleatorio en cada iteración guardándolo en la lista *l*.

2. Abre ahora el programa llamado PIA_S02_simple_patterns.py.

   2.1.Ejecuta el fichero desde spyder y comenta qué problemas te ha dado.

   Según la terminal de Spyder, indica que el índice de la lista está fuera del rango. Es decir, que se esta intentando acceder a un elemento de la lista que no existe.

3. Este programa requiere de parámetros para poder funcionar. El programa busca las ocurrencias de una palabra en un texto en un fichero. Tanto el fichero, como la palabra a buscar se reciben a través de la interfaz de comandos. Por ejemplo,

```
./02_simple_patterns.py texto.txt Mancha
The word Mancha appeared 1 times in file texto.txt
```

   Puedes ejecutarlo con parámetros desde Spyder usando el comando runfile de la consola de Spyder para ejecutar el texto:

```
runfile('ruta_dir/02_simple_patterns.py',wdir=' ruta_dir',args='texto.txt Mancha')
```

   3.1.¿Te parece apropiado este tratamiento de textos para un entorno de simulación? ¿Y para un entorno de producción?

   Tanto para un entorno de simulación como de producción lo veo adecuado. Ya que permite saber si la palabra que se quiere buscar se encuentra en el fichero o no sin detener la ejecución del programa.

4. Abre ahora los ficheros 02_simple_patterns_error_control.py y 02_simple_patterns_error_control.c.

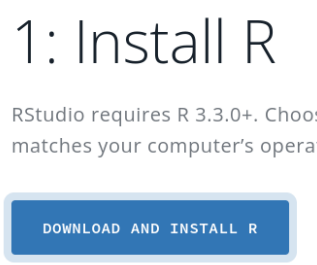
4.1.compáralos rellenando esta tabla comentando qué te parece cada lenguaje desde el punto de vista de la simplicidad, la legibilidad y la capacidad de prototipado:

Lenguaje	Simplicidad	Legibilidad	Capacidad de prototipado
C	Complejo. Utiliza bastantes caracteres como corchetes y paréntesis en una sola línea	Menor. Hace que el código se vea menos simple al no utilizar metáforas	No es un lenguaje adecuado para prototipado porque carece de simplicidad
Python	Simple. En vez de utilizar corchetes y otros elementos utiliza las tabulaciones	Mayor. Hace que el código sea más simple usando metáforas	Lenguaje adecuado para prototipado. Ya que la simplicidad y el pragmatismo son esenciales para esto

(puedes leer las secciones 5,6 y 7 de la UT1 para orientarte)

4.2. ¿Qué control de errores implementa el programa?

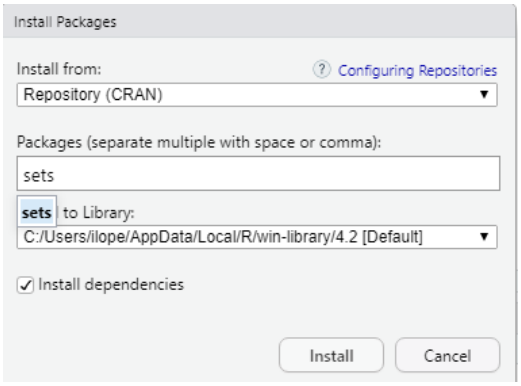
Instala R y RStudio https://www.rstudio.com/products/rstudio/download/



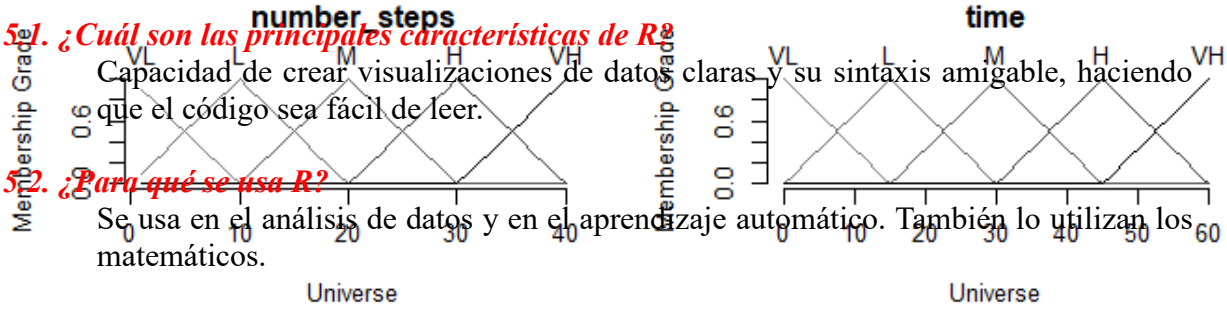
Instala RTools, para cargar librerías desde código fuente: https://cran.rstudio.com/bin/windows/Rtools/rtools42/rtools.html

Abre ahora RStudio, vamos a ejecutar un programa en lenguaje R. El programa que vamos a ejecutar necesita la librería sets de R.

Ahora instala la librería sets desde la opción Tools -> Install Packages:



5. Responde a las siguientes preguntas basándote en las secciones 6 y 7 de la UT1:



5.1. ¿Cuál son las principales características de R?

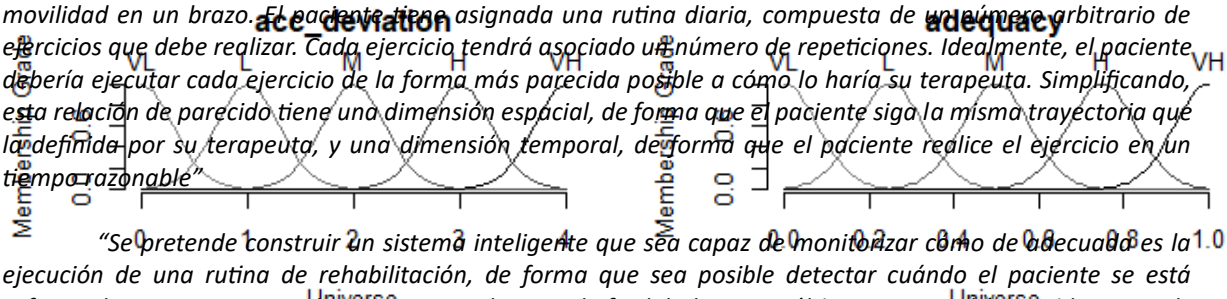
Capacidad de crear visualizaciones de datos claras y su sintaxis amigable, haciendo que el código sea fácil de leer.

5.2. ¿Para qué se usa R?

Se usa en el análisis de datos y en el aprendizaje automático. También lo utilizan los matemáticos.

Ejecución de un script en R:

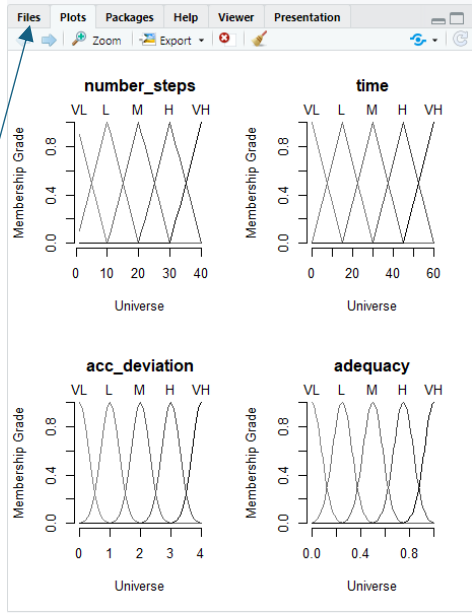
"Suponga una terapia de rehabilitación física donde un paciente tiene como objetivo recuperar la movilidad en un brazo. El paciente tiene asignada una rutina diaria, compuesta de un número arbitrario de ejercicios que debe realizar. Cada ejercicio tendrá asociado un número de repeticiones. Idealmente, el paciente debería ejecutar cada ejercicio de la forma más parecida posible a cómo lo haría su terapeuta. Simplificando, esta relación de parecido tiene una dimensión espacial, de forma que el paciente siga la misma trayectoria que la definida por su terapeuta, y una dimensión temporal, de forma que el paciente realice el ejercicio en un tiempo razonable"



"Se pretende construir un sistema inteligente que sea capaz de monitorizar cómo de adecuada es la ejecución de una rutina de rehabilitación, de forma que sea posible detectar cuándo el paciente se está esforzando en exceso o, por el contrario, con demasiada facilidad. En este último caso, tendría sentido asignarle un ejercicio, o rutina, más ambicioso que fomente la recuperación de movilidad en el miembro a rehabilitar."

*"Para representar tanto las variables que conforman el sistema como su núcleo de inferencia, se opta por utilizar **un sistema basado en lógica difusa**"*

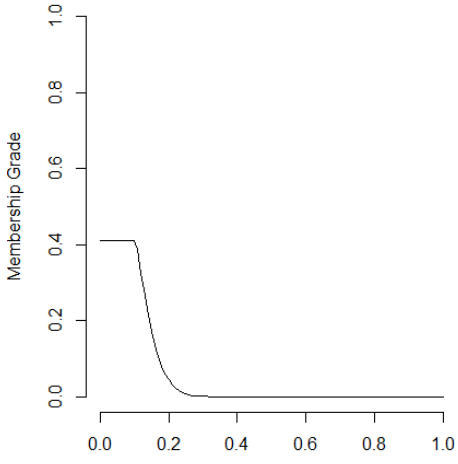
El fichero Fuzzy_PhyReUp.R de la carpeta S03_R_Sample contiene el código con el ejemplo del sistema de lógica difusa, ejecútalo y contesta a las siguientes preguntas:

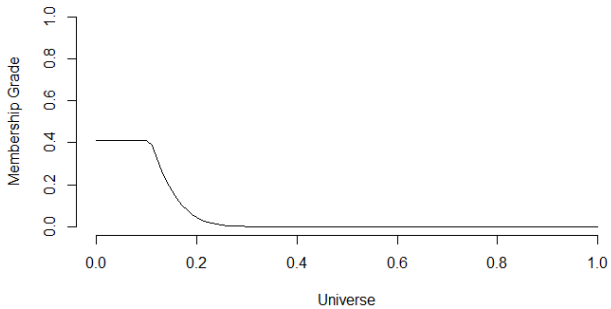


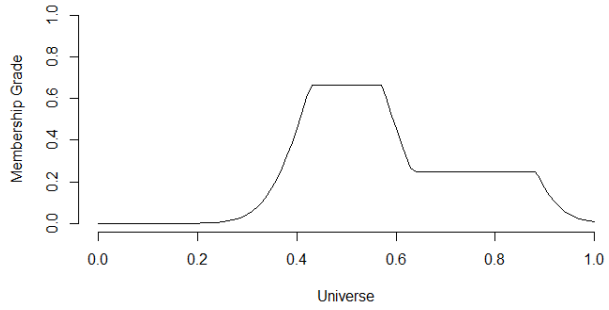
5.3. ¿En qué consiste la lógica difusa?

Su naturaleza es una lógica multivaluada y puede tener valores en el intervalo [0, 1], es decir representan grados de verdad. Se usa para modelar información imprecisa o ambigua como los diagnósticos de enfermedades.

5.4. Muestra los gráficos que ha generado el programa en R. (Puedes hacerlo desde el panel "Plots")







Teniendo en cuenta que en las particiones difusas las siglas VL (*very low*), L (*low*), M (*medium*), H (*high*) y VH (*very high*) representan valores de salida "etiquetados" fíjate en los ejemplos de muestra que procesa el modelo creado y en los valores de la partición difusa **idoneidad** "adequacy".

Si el resultado de la inferencia es de 0.07 para un paciente es "número de pasos 8, tiempo=30 y desviación 3,6":

5.5. ¿Qué resultado de idoneidad obtiene el primer paciente "example.1"?

[Pista: Compara la gráfica resultante con la gráfica de la partición difusa "adequacy"]

```
> example.1 <- fuzzy_inference(model, list(number_steps = 8,
+                                          time = 30, acc_deviation = 3.6))
> gset_defuzzify(example.1, "centroid")
[1] 0.07735396
> plot(example.1)
```

Para terminar, examina y ejecuta el código de la carpeta S03_Rock_Paper_Scissors donde encontrarás 5 ficheros con código Python en una versión básica "basic", una versión con control de errores "Error_control", una con código más escalable "clean code" y dos con versiones muy básicas de acciones "inteligentes" (Weak) "Basic IA"y "More IA".

Responde a las siguientes preguntas:

5.6. ¿Qué significa weak IA?

Es una IA que realiza tareas específicas de forma muy eficaz sin conciencia o capacidad de razonamiento como un humano. Sirven para resolver problemas o realizar tareas específicas y no pueden hacer otras tareas de las cuales no han sido diseñadas para tal.

5.7. ¿Qué es una heurística?

Es una técnica o estrategia que facilita la toma de decisiones, la resolución de problemas o el descubrimiento de soluciones. Puede producir una solución aceptable en un tiempo razonable. Útiles en situaciones donde una solución perfecta es impracticable en tiempo o costo.

EVALUACIÓN DE LA PRÁCTICA:

Se evaluarán las respuestas con una nota alfanumérica:
A= sobresaliente [10-9]
B= notable [8-7]
C= suficiente [6-5]
D= insuficiente [<5]
E= no entregado