Prácticas de Autómatas y Lenguajes. Curso 2023/24
Práctica 0: Expresiones regulares

Duración: 1 semana.
Entrega: Semana del 2 de octubre, cada grupo antes de su clase.
Peso: 10% de la nota de prácticas.

Descripción del enunciado:

En esta práctica diseñaremos expresiones regulares y usaremos el módulo re (*regular expression operations*) de Python para comprobarlas. Una expresión regular es una cadena de caracteres que se utiliza para describir o encontrar patrones dentro de otras cadenas, en base al uso de delimitadores y ciertas reglas de sintaxis. Conviene que estudies el contenido de la presentación Expresiones Regulares disponible en moodle antes de afrontar esta práctica.

Es importante que tengas en cuenta que las expresiones regulares usadas habitualmente incluyen una sintaxis más rica que la utilizada en clase de teoría. Es recomendable por tanto que estudies la sintaxis de las expresiones regulares utilizadas en el módulo re de Python y atiendas a las explicaciones de tu profesor de prácticas. En el siguiente enlace tienes un tutorial alternativo: https://docs.python.org/3/howto/regex.html

Como es probable que no hayas utilizado el lenguaje de programación Python con anterioridad, a continuación te facilitamos algunos enlaces a tutoriales que te pueden resultar útiles. No obstante para la realización de esta primera práctica no es necesario programar, sino simplemente diseñar las expresiones regulares que te pedimos.

Tutoriales de python:

Documentación oficial:

- https://docs.python.org/3/tutorial/ (en inglés)
- https://docs.python.org/es/3/tutorial/ (en español)

Lo más básico de Python para programadores de C:

- https://engineering.purdue.edu/~milind/datascience/2018spring/notes/lecture-2.pdf

Tutoriales de Python más detallados:

- https://anandology.com/python-practice-book/getting-started.html
- https://inst.eecs.berkeley.edu/~cs188/fa20/project0/

Guías de estilo:

- https://gist.github.com/kamikaze-lab/df2b4df198605abad846 (en español)
- https://www.python.org/dev/peps/pep-0008/ (en inglés)

Uso de conda:

- https://docs.conda.io/projects/conda/en/4.6.0/_downloads/52a95608c49671267e40c689e0bc00ca/conda-cheatsheet.pdf
- https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

Instrucciones:

Edita el fichero *regular_expressions.py* y completa las expresiones regulares para las variables `RE1` a `RE6` de acuerdo a las descripciones de los ejercicios siguientes. La expresión `RE0` se facilita a modo de ejemplo. Puedes probar tus expresiones regulares ejecutando el código *test_p0.py*, que contiene varios ejemplos.

Ejercicio 0:

Diseña una expresión regular sobre el alfabeto {a,b} que represente las cadenas no vacías que terminan en a. Asigna la expresión regular a la variable RE0.

Ejercicio 1 (1 punto):

Diseña una expresión regular para el siguiente lenguaje:

L = {w ∈ {a,b}*: w no contiene la cadena ba}

Ejemplo de cadenas validas: a, ab, aab

Ejemplo de cadenas invalidas: ba, aba

Asigna la expresión regular a la variable RE1.

Ejercicio 2 (1 punto):

Diseña una expresión regular para el lenguaje sobre el alfabeto {a,b} que no contiene 2 símbolos idénticos y consecutivos, es decir, el símbolo en la posición i debe ser diferente al símbolo en la posición i+1. Asigna la expresión regular a la variable RE2.

Nota: Se debe rechazar la cadena vacía.

Ejemplo de cadenas válidas: a, ab, aba
Ejemplo de cadenas inválidas: aa, bb, aaa

Ejercicio 3 (2 puntos):

Diseña una expresión regular para el siguiente lenguaje:

$L = \{w \in \{a,b\}^* : \text{las subcadenas aa y bb son parte de w}\}$

Ejemplo de cadenas validas: aabb, abbaa

Ejemplo de cadenas invalidas: ababaa, abbabb

Asigna la expresión regular a la variable RE3.

Ejercicio 4 (2 puntos):

Diseña una expresión regular que acepte las cadenas no vacías que representan los números enteros del 0 al 255. La expresión regular debe rechazar las cadenas con las siguientes características:

- Ceros innecesarios a la izquierda.
- Símbolos que no están en el alfabeto.

Ejemplo de cadenas validas: 0, 10, 255, 2, 20.

Ejemplo de cadenas invalidas: -1, +1, 3.33, 007, 256.

Asigna la expresión regular a la variable RE4.

Ejercicio 5 (2 puntos):

Diseña una expresión regular para el siguiente lenguaje:

L = {w ∈ {a,b}*: w no contiene más de dos apariciones de la subcadena aa}

Ejemplo de cadenas validas: a, aa, abaa, abaab, abbaa

Ejemplo de cadenas invalidas: aabaaaa, bbaabaabaa

Asigna la expresión regular a la variable RE5.

Ejercicio 6 (2 puntos)

Diseña una expresión regular para el siguiente lenguaje:

L = {w ∈ {1,0}*: w tiene como máximo un par de ceros consecutivos y como máximo un par de unos consecutivos}

Ejemplo de cadenas validas: 0, 1, 00, 001, 0011, 00110

Ejemplo de cadenas invalidas: 0000, 1111, 001100

Asigna la expresión regular a la variable RE6.

Normas de entrega:

La entrega la realizará sólo uno de los miembros de la pareja a través de la tarea disponible en Moodle.

Sólo se debe entregar el fichero *regular_expressions.py* con las expresiones regulares pedidas en los ejercicios 1-6.