



# UNIVERSIDAD AUTÓNOMA DE CHIAPAS

---

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

INGENIERÍA EN DESARROLLO Y TECNOLOGÍAS DE SOFTWARE

## TITULO DE ACTIVIDAD

Actividad 2. Ejercicios.

## NOMBRE, GRADO Y GRUPO DEL ALUMNO

Mónica Isabel López Flores del 6ºM

## NUMERO DE CONTROL DEL ALUMNO

A210525.

## MATERIA

Compiladores.

## DOCENTE

Luis Gutiérrez Alfaro.



Sábado 27 de enero de 2024.

Tuxtla Gutiérrez Chiapas, México.

## **Contenido**

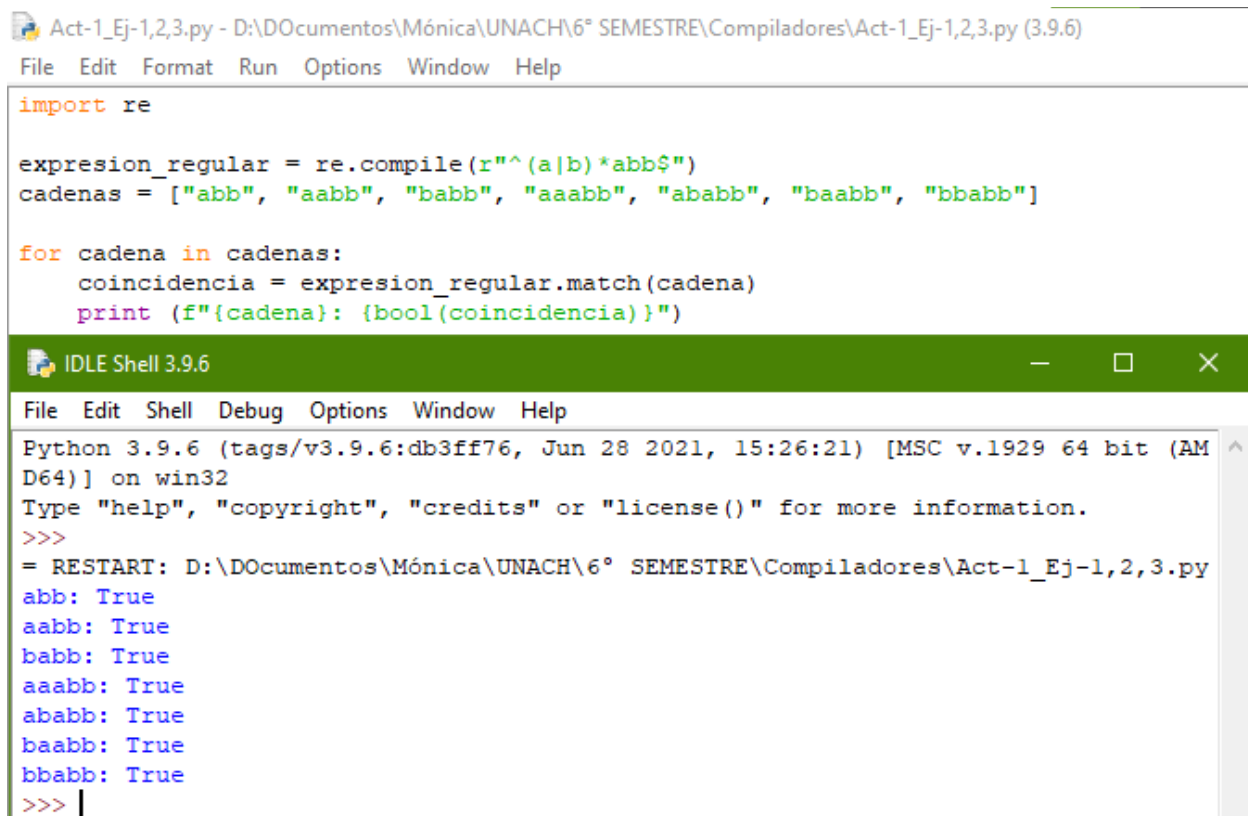
Ejercicio 1 .....	2
Ejercicio 2.....	2
Ejercicio 3.....	3
Ejercicio 4.....	4
Ejercicio 5.....	5

## Definir El Concepto De Expresión Regular.

Instrucciones: Realice los siguientes ejercicios

### Ejercicio 1

Realice una expresión regular de todas las cadenas con símbolos a y b, que terminan con el sufijo abb, Ejemplo de estas cadenas son: abb, aabb, babb aaabb, ababb, baabb bbabb



The image shows a screenshot of a Python script and its execution. The script is named 'Act-1\_Ej-1,2,3.py' and is located at 'D:\Documentos\Mónica\UNACH\6° SEMESTRE\Compiladores\Act-1\_Ej-1,2,3.py (3.9.6)'. The script imports the 're' module and defines a regular expression 'expresion\_regular = re.compile(r"^(a|b)\*abb\$")'. It then defines a list of strings 'cadenas' containing 'abb', 'aabb', 'babb', 'aaabb', 'ababb', 'baabb', and 'bbabb'. A loop iterates over each string in 'cadenas', checks if it matches the regular expression using 'expresion\_regular.match(cadena)', and prints the result in the format '{cadena}: {bool(coincidencia)}'.

```
import re

expresion_regular = re.compile(r"^(a|b)*abb$")
cadenas = ["abb", "aabb", "babb", "aaabb", "ababb", "baabb", "bbabb"]

for cadena in cadenas:
    coincidencia = expresion_regular.match(cadena)
    print (f"{cadena}: {bool(coincidencia)}")
```

The execution output in the IDLE Shell 3.9.6 window shows the following results:

```
Python 3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: D:\Documentos\Mónica\UNACH\6° SEMESTRE\Compiladores\Act-1_Ej-1,2,3.py
abb: True
aabb: True
babb: True
aaabb: True
ababb: True
baabb: True
bbabb: True
>>> |
```

### Ejercicio 2

Realice una expresión regular de todas las cadenas de con simbolos o y 1, que primero tengan los simbolos 1's con longitud impar y después aparezcan los O's con longitud par. Ejemplo de estas cadenas son 100, 10000, 1000000, 111001110000111110000

```
import re

expresion_regular = re.compile(r"^1(11)*(00)+")
cadenas = ["100", "10000", "1000000", "11100", "1110000", "111110000"]

for cadena in cadenas:
    coincidencia = expresion_regular.match(cadena)
    print(f"{cadena}: {bool(coincidencia)}")
```

```
IDLE Shell 3.9.6
File Edit Shell Debug Options Window Help
Python 3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: D:\Documentos\Mónica\UNACH\6º SEMESTRE\Compiladores\Act-1_Ej-1,2,3.py
100: True
10000: True
1000000: True
11100: True
1110000: True
111110000: True
>>> |
```

### Ejercicio 3

Para la expresión regular  $(+/-)?d + d +$  indique las cadenas correctas de los siguientes incisos (Nota En esta expresión  $\epsilon$  es un símbolo no el operador concatenación y  $d$  representa los dígitos del 0 al 9).

a) -20 43

b) 0.3216

c) 329.

d) 217 92

e) +2019

1)762

4)-4555

```
import re

expresion_regular = re.compile(r"^[+-]?\d+\.\d+$")
cadenas = ["-20.43", "0.3216", "329.", "217.92", "+2019", "+.762", "-.4555"]

for cadena in cadenas:
    coincidencia = expresion_regular.match(cadena)
    print(f"{cadena}: {bool(coincidencia)}")
```

IDLE Shell 3.9.6

File Edit Shell Debug Options Window Help

Python 3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license()" for more information.

>>>

= RESTART: D:\Documentos\Mónica\UNACH\6° SEMESTRE\Compiladores\Act-1\_Ej-1,2,3.py

-20.43: True

0.3216: True

329.: False

217.92: True

+2019: False

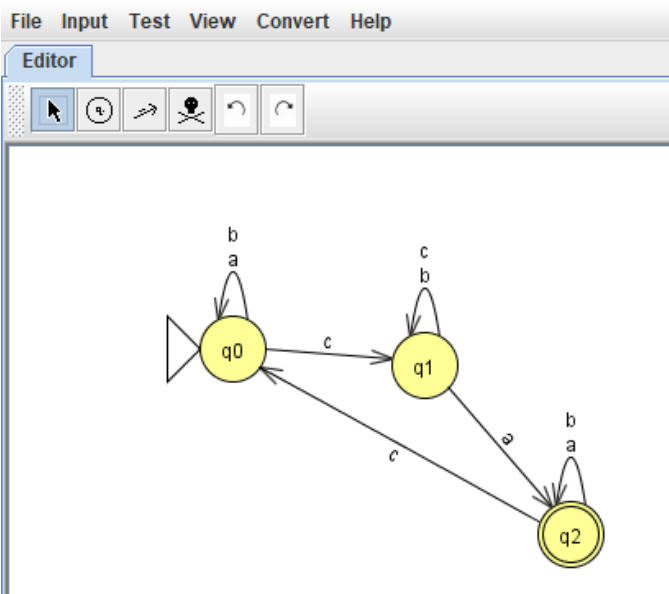
+.762: False

-.4555: False

>>> |

## Ejercicio 4

Obtenga un AFD dado el siguiente lenguaje definido en el alfabeto  $\Sigma=\{a,b,c\}$ . El conjunto de cadenas que inician en la sub-cadena "ac" y terminan en la sub-cadena "ab".



**Ejercicio 5**

Obtenga un AFND dado el siguiente lenguaje definido en el alfabeto  $\Sigma=(abc)$ . El conjunto de cadenas que no inician en la sub-cadena "ac" o no terminan en la sub cadena "ab".

