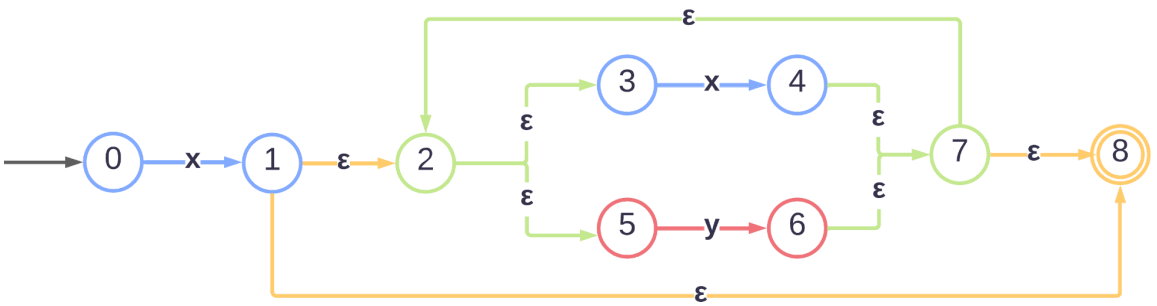
Trabajo Práctico nro: 2

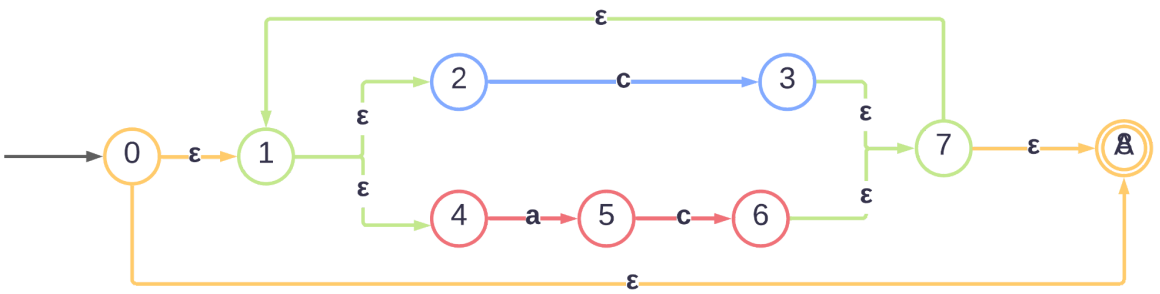
Grupo: Franco Bertoldi, Francisco Lopez Garcia, Lucas Garcia

Ejercicio 1

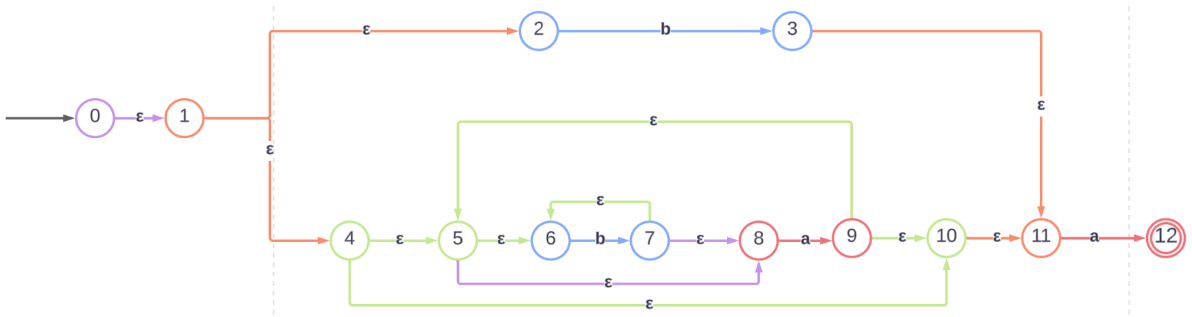
a)



b)

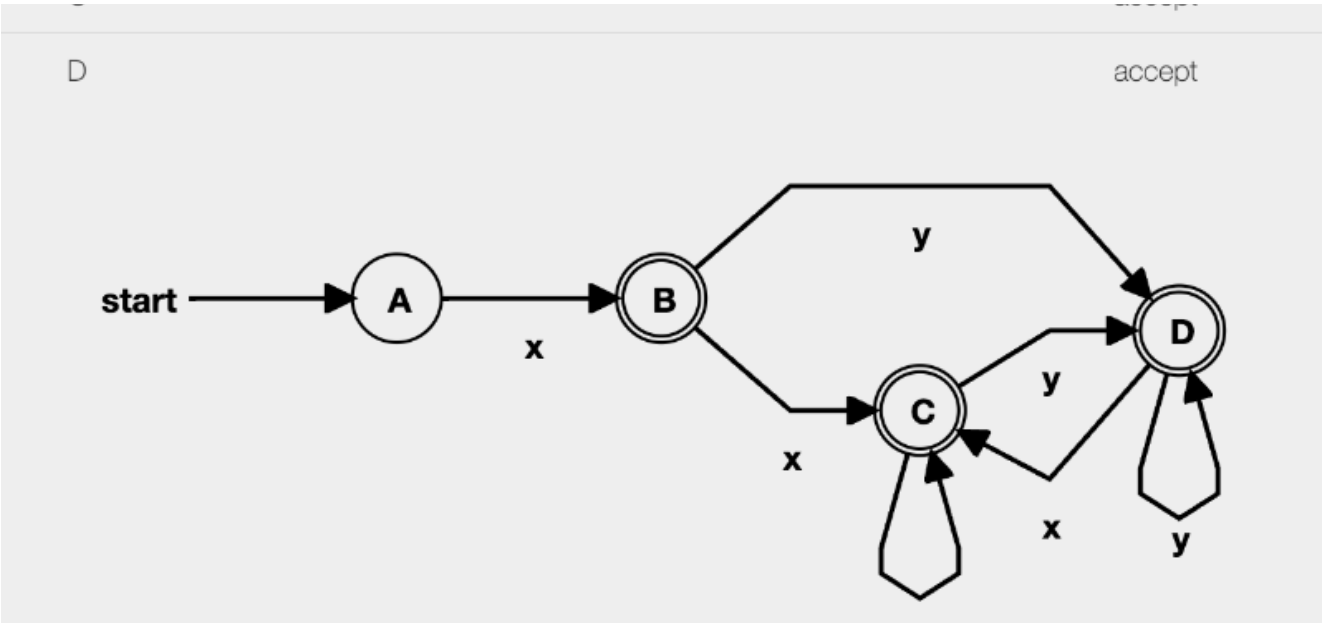


c)



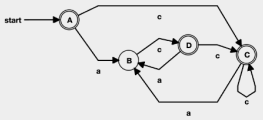
Ejercicio 2

a)



b)

NFA STATE	DFA STATE	TYPE	a	c
{0,1,2,4,8}	A	accept	B	C
{5}	B			D
{1,2,3,4,7,8}	C	accept	B	C
{1,2,4,6,7,8}	D	accept	B	C



c)

- {0} = **{0, 1, 2, 4, 5, 6, 8, 10, 11}**
- {0, 1, 2, 4, 5, 6, 8, 10, 11}a = {9, 12}
- {0, 1, 2, 4, 5, 6, 8, 10, 11}b = {3, 7}

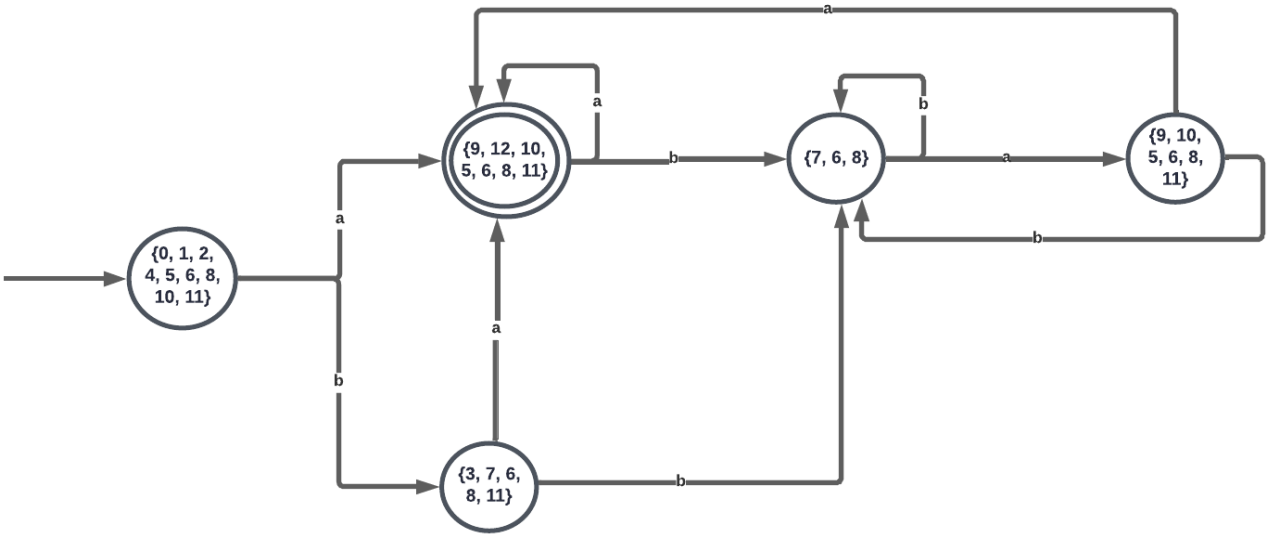
- {9, 12} = **{9, 12, 10, 5, 6, 8, 11}**
- {9, 12, 10, 5, 6, 8, 11}a = {9, 12}
- {9, 12, 10, 5, 6, 8, 11}b = {7}

- {3, 7} = **{3, 7, 6, 8, 11}**
- {3, 7, 6, 8, 11}a = {9, 12}
- {3, 7, 6, 8, 11}b = {7}

- {7} = **{7, 6, 8}**
- {7, 6, 8}a = {9}
- {7, 6, 8}b = {7}

- {9} = **{9, 10, 5, 6, 8, 11}**
- {9, 10, 5, 6, 8, 11}a = {9, 12}
- {9, 10, 5, 6, 8, 11}b = {7}

c)



Parte B

Ejercicio 1

(c|ab)* (a|cc)*

El lenguaje generado por la expresión regular (ca|b)(a|cc) es aquel que acepta cualquier cadena de texto iniciada con “ca” o “b” y terminada en “a” o “cc” pudiendo repetir cualquiera de estas las veces que haga falta

(0|1|.. .|9|A|B|C|D|E|F)(w|W)

El lenguaje generado por la expresión regular (0|1|.. .|9|A|B|C|D|E|F)(w|W) es aquel que acepta cualquier cadena de texto iniciada con un dígito hexadecimal “(0-9,A-F)” y terminada en “w” o “W”

(A|B|…|Z)(a|b|…z)*

El lenguaje generado por la expresión regular (A|B|…|Z)(a|b|…z)* es aquel que acepta cualquier cadena de texto iniciada con una letra mayúscula “”(A-Z)” y seguida de letras minusculas “(a-z)

Ejercicio 2

1. Fecha con formato dd/mm/yyyy o dd-mm-yyyy.

   r'^(0[1-9] | [1-2][0-9] | 3[0-1])[-/](0[1-9] | 1[0-2])[-/]\d{4}$'

2. Número real con dos decimales y separador de miles.

   r'^\d{1,3}(,\d{3})*(\.\d{2})?$'

3. ID de un vídeo de Youtube.

   r'^[a-zA-Z0-9_-]{11}$'

4. Cuenta de Email de alumno de la Universidad de Mendoza.

   r'^[a-zA-Z0-9._%+-]+@[alumno]\.[um]\.[edu]\.[ar]$'

5. Número de teléfono móvil de Argentina, que incluya código de país, de provincia, y el 15.

   r'^[15][0-9]{3}[0-9]{7}$'

6. CUIL.

   r'^[0-9]{2}[0-9]{6,8}[0-9]$'

7. Seguridad de una contraseña, que incluya:

   A. Que contenga al menos un número.
   B. Que contenga al menos una letra mayúscula.
   C. Que contenga al menos carácter especial.
   D. La longitud sea como mínimo 8 caracteres.

r'^(?=.*\d)(?=.*[A-Z])(?=.*[!@#$%^&*()_+{}|:"<>?`~\-=[\]\\;,./])(?=.{8,16})\S+$'

Ejercicio 3

```
import re


def validar_fecha(fecha):
    patron =
re.compile(r'^(0[1-9]|[12][0-9]|3[01])[-/](0[1-9]|1[0-2])[-/]\d{4}$')
    return patron.match(fecha)


def validar_numero_real(numero):
    patron = re.compile(r'^\d{1,3}(,\d{3})*(\.\d{2})?$')
    return patron.match(numero)


def validar_id_youtube(id):
    patron = re.compile(r'^[a-zA-Z0-9_-]{11}$')
    return patron.match(id)


def validar_cuenta_email_alumno_um(email):
    patron =
re.compile(r'^[a-zA-Z0-9._%+-]+@[alumno]\.[um]\.[edu]\.[ar]$')
    return patron.match(email)


def validar_numero_tel_arg(telefono):
    patron = re.compile(r'^[15][0-9]{3}[0-9]{7}$')
    return patron.match(telefono)


def validar_cuil(cuil):
    patron = re.compile(r'^[0-9]{2}[0-9]{6,8}[0-9]$')
    return patron.match(cuil)


def validar_contrasena_segura(contrasena):
```

```
    patron =
re.compile(r'^(?=.*\d)(?=.*[A-Z])(?=.*[!@#$%^&*()_+{}|:"<>?`~\-=[\]\\;,
./])(?=.{8,16})\S+$')
    return patron.match(contrasena)
```