Universidad del Valle de Guatemala
Lógica Matemática S30

Juan Miguel González-Campo 21077
Pedro Marronquín 21801
Raúl Sánchez 21401
Mariel Guamuche 21150

Proyecto 1

I. Descripción del juego de datos

El set de datos cuenta con 15 columnas (Identifier, Edition Statement, Place of Publication, Date of Publication, Publisher, Title, Author, Contributors, Corporate Author, Corporate Contributors, Former owner, Engraver, Issuance type, Flickr URL, Shelfmarks) con un total de 8287 observaciones o filas (incluyendo una fila de los encabezados).
Cabe destacar que los tipos de datos son string[1] (Edition Statement, Place of Publication, Data of Publication, Publisher, Title, Author, Contributors, Former Owner, Issuance type, Flickr URL y Shelfmarks), int (identifier) y float[2] (Corporate Author, Corporate Contributors, Engraver), en donde estos últimos son celdas en el csv vacías.

La figura 2 contiene un resumen del total de datos en null o NaN dentro del set de datos. Debido a estos datos, más adelante se realiza un ajuste en la lectura de los datos para facilitar la creación de las expresiones regulares.

Figura 1.
Descripción general del set de datos "BL-Flickr-Images-Book.csv"

```
Tipos de los datos:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8287 entries, 0 to 8286
Data columns (total 15 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Identifier             8287 non-null   int64
 1   Edition Statement      773 non-null    object
 2   Place of Publication   8287 non-null   object
 3   Date of Publication    8106 non-null   object
 4   Publisher              4092 non-null   object
 5   Title                  8287 non-null   object
 6   Author                 6509 non-null   object
 7   Contributors           8287 non-null   object
 8   Corporate Author       0 non-null      float64
 9   Corporate Contributors 0 non-null      float64
 10  Former owner           1 non-null      object
 11  Engraver               0 non-null      float64
 12  Issuance type          8287 non-null   object
 13  Flickr URL             8287 non-null   object
 14  Shelfmarks             8287 non-null   object
dtypes: float64(3), int64(1), object(11)
memory usage: 971.3+ KB
None
Información del dataframe completo:

(8287, 15)
```

[1] La librería de pandas coloca las cadenas de caracteres (strings) como tipo object.
[2] La librería de pandas define como NaN (Not a Number) las celdas vacías de un archivo csv.

Figura 2.

Descripción del total de datos nulos o NaN en el set de datos "BL-Flickr-Images-Book.csv"

```
Información de total datos nulos o NaN:

Identifier                 0
Edition Statement       7514
Place of Publication       0
Date of Publication      181
Publisher               4195
Title                      0
Author                  1778
Contributors               0
Corporate Author        8287
Corporate Contributors  8287
Former owner            8286
Engraver                8287
Issuance type              0
Flickr URL                 0
Shelfmarks                 0
```

II. Expresiones regulares definidas

A. Encabezados

En cuanto a la obtención de los encabezados, se tomó por default la primera fila del set de datos, en donde se hizo el uso de la función split de la librería re (figura 3). El patrón para la separación de valores es que se encontrase una coma (figura 4).

Figura 3.

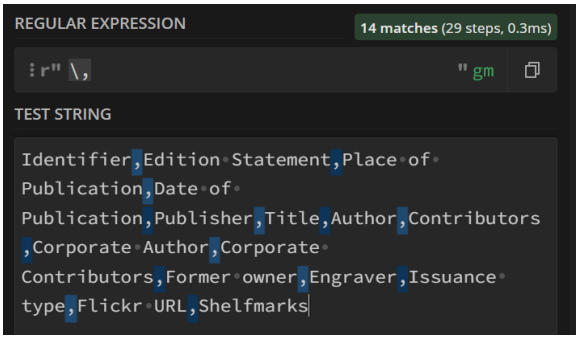
Código para encabezados

```
def get_encabezados(d):
    linea_encabezados = d[0]
    temp = []

    temp = re.split("\,", linea_encabezados)
    for i in range(len(temp)):
        temp[i] = temp[i].strip("\n")
    return temp
```

Figura 4.

Evaluación de expresión regular en python



Se hizo uso de la herramienta https://regex101.com/ para la evaluación de la expresión regular

B. Contenido por columna

Como se había mencionado anteriormente para aquellas celdas vacías al momento de ser leídas con la función definida, estas estarán indicadas con la separación de comas. Esto presenta una dificultad adicional, por lo cual se definió una función para la preparación de los datos (figura 5). Así mismo, durante la lectura de datos, en la columna Identifier se agregan cero al inicio, los cuales fueron sustituidos por un carácter vacío haciendo uso de la función sub. El funcionamiento de las expresiones utilizadas en las sustituciones pueden ser consultadas en las figuras 6 y 7.

Figura 5.

Código de la función coma_seguido_coma

```
def coma_seguido_coma(texto):
  temp = re.sub(r'((?<=\,)(?=,))'," ", texto) # agrega dos espacios en los
casos de comas seguidas
  temp = re.sub(r'^0{6}|^0{5}|^0{4}|^0{3}|^0{2}|^0{1}',"", temp) # elimina los 0
previos en el identifier
  return temp
```

Figura 6.

Evaluación de expresión regular en python

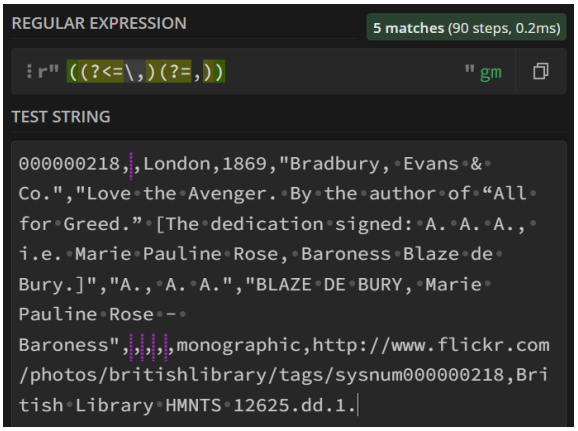
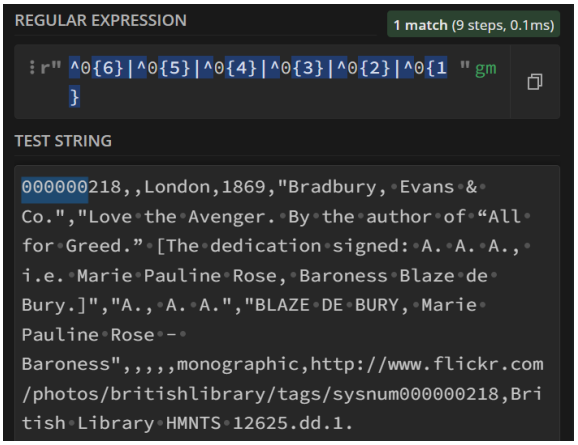


Figura 7.
Evaluación de expresión regular en python



A continuación (figura 8, 9 y 10) se definen y verifican el funcionamiento de la definición de las expresiones regulares creadas. Recordando que el orden de procedencia se realiza de izquierda a derecha. La función parse_linea asigna la búsqueda de matches con la expresión regular a una lista, la cual es limpiada de datos extras agregados durante la etapa de lectura de datos. La expresión utilizada evalúa 6 casos diferentes. Al no haberse producido error en la ejecución del código, este despliega un mensaje de su correcta ejecución (figura 10). Para finalizar el proceso, se crea un archivo csv con el *dataframe* generado.

Figura 8.
Código de parse_linea

```
def parse_linea(l):

    temp =
re.findall(r'\"[^\"]*(?:\"\"[^\"]*)*\",|[\"].*\",|[\"][^\"]*[\"]\n|[^\,]*[\,]|.+
\n|.+', l)    #Regex de lectura CSV

    temp[-1] = temp[-1].strip("\n") #Eliminamos caracteres de nueva linea

    for i in range(len(temp)): #Eliminamos las comas y las comillas del texto
        temp[i] = re.sub(r',$|["]','',temp[i])
        temp[i] = temp[i].strip(" ")

    return temp
```

Figura 9.

Evaluación de expresión regular en python

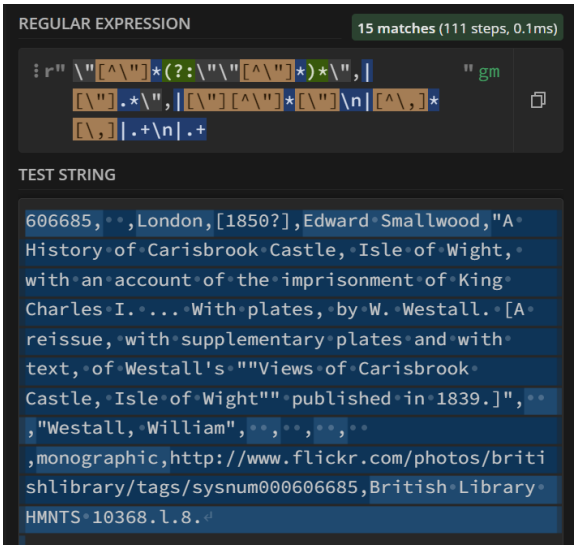


Figura 10.

Conversión correcta de los datos en un dataframe

```
Inicio de datos:
  Identifier            Edition Statement  ...                                          Flickr URL                                  Shelfmarks
0        206                          NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...    British Library HMNTS 12641.b.30.
1        216                          NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...    British Library HMNTS 12626.cc.2.
2        218                          NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...    British Library HMNTS 12625.dd.1.
3        472                          NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...  British Library HMNTS 10369.bbb.15.
4        480  A new edition, revised, etc.  ...  http://www.flickr.com/photos/britishlibrary/ta...     British Library HMNTS 9007.d.28.

[5 rows x 15 columns]

Fin de datos:
     Identifier Edition Statement  ...                                          Flickr URL                                  Shelfmarks
8282    4158088              NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...  British Library HMNTS|British Library HMNTS 10...
8283    4158128              NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...  British Library HMNTS|British Library HMNTS 10...
8284    4159563              NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...  British Library HMNTS|British Library HMNTS 19...
8285    4159587              NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...  British Library HMNTS|British Library HMNTS 10...
8286    4160339              NaN  ...  http://www.flickr.com/photos/britishlibrary/ta...  British Library HMNTS|British Library HMNTS 79...

[5 rows x 15 columns]

 CSV exportado!
```

Finalmente con este data frame podemos comparar con el data frame que se genera cuando importamos directamente a Pandas el archivo CSV original.

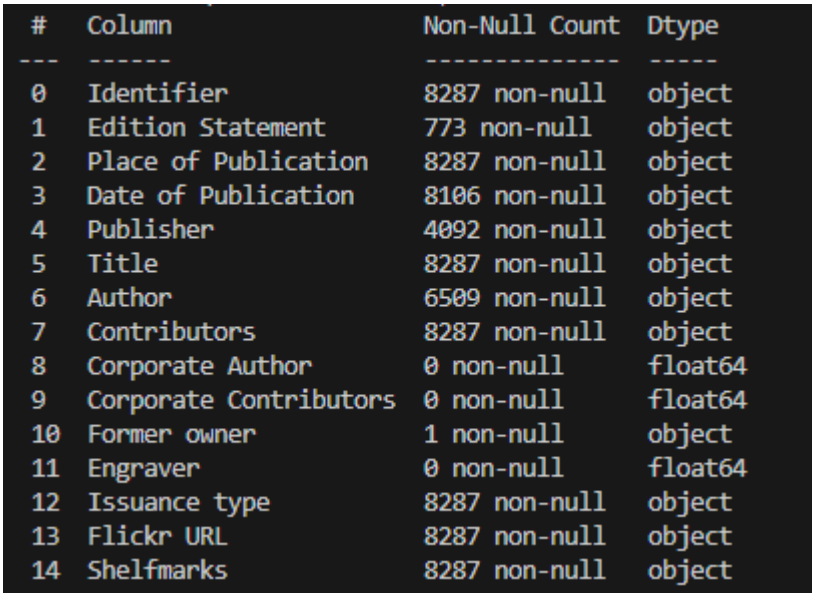
Figura 11.

Información de data frame generado con expresión regular.

```
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Identifier           8287 non-null   int64
 1   Edition Statement    773 non-null    object
 2   Place of Publication 8287 non-null   object
 3   Date of Publication  8106 non-null   object
 4   Publisher            4092 non-null   object
 5   Title                8287 non-null   object
 6   Author               6509 non-null   object
 7   Contributors         8287 non-null   object
 8   Corporate Author     0 non-null      float64
 9   Corporate Contributors 0 non-null    float64
 10  Former owner         1 non-null      object
 11  Engraver             0 non-null      float64
 12  Issuance type        8287 non-null   object
 13  Flickr URL           8287 non-null   object
 14  Shelfmarks           8287 non-null   object
```

Figura 12.

Información de data frame generado por archivo CSV original.



#	Column	Non-Null Count	Dtype
0	Identifier	8287 non-null	object
1	Edition Statement	773 non-null	object
2	Place of Publication	8287 non-null	object
3	Date of Publication	8106 non-null	object
4	Publisher	4092 non-null	object
5	Title	8287 non-null	object
6	Author	6509 non-null	object
7	Contributors	8287 non-null	object
8	Corporate Author	0 non-null	float64
9	Corporate Contributors	0 non-null	float64
10	Former owner	1 non-null	object
11	Engraver	0 non-null	float64
12	Issuance type	8287 non-null	object
13	Flickr URL	8287 non-null	object
14	Shelfmarks	8287 non-null	object

Aquí podemos observar que ambos data frames tienen la misma cantidad de objetos no nulos (y por lo tanto la misma cantidad de objetos nulos) lo cual nos indica que nuestra expresión regular si leyó correctamente los datos, lo cual se puede confirmar en una exploración del archivo CSV “dataframe.csv” que generamos con el data frame generado con nuestra expresión regular y comparando con el CSV original.

III. Diagramas de las máquinas deterministas

- ➔ $M_1 = (Q, S_0, I, O, f, g)$,
  - $Q = \{1, 2, 3, 4, 5\}$
  - $S_0 = \{1\}$
  - $I = \{"[^"] * (?:""[^"] *) * ", \}$
  - $O = \{0, 1\}$
  - $f: Q \times I \to Q$
  - $g: Q \to O$

- ➔ $M_2 = (Q, S_0, I, O, f, g)$,
  - $Q = \{1, 2, 3, 4\}$
  - $S_0 = \{1\}$
  - $I = \{["].* ",\}$
  - $O = \{0, 1\}$
  - $f: Q \times I \to Q$
  - $g: Q \to O$

- ➔ $M_3 = (Q, S_0, I, O, f, g)$,
  - $Q = \{1, 2, 3, 4\}$
  - $S_0 = \{1\}$
  - $I = \{["][^"] * ["]\backslash n\}$
  - $O = \{0, 1\}$
  - $f: Q \times I \to Q$
  - $g: Q \to O$

- ➔ $M_4 = (Q, S_0, I, O, f, g)$,
  - $Q = \{1, 2\}$
  - $S_0 = \{1\}$
  - $I = \{[^,] * [,]\}$
  - $O = \{0, 1\}$
  - $f: Q \times I \to Q$
  - $g: Q \to O$

- ➔ $M_5 = (Q, S_0, I, O, f, g)$,
  - $Q = \{1, 2\}$
  - $S_0 = \{1\}$
  - $I = \{. + \backslash n\}$
  - $O = \{0, 1\}$
  - $f: Q \times I \to Q$
  - $g: Q \to O$

- ➔ $M_6 = (Q, S_0, I, O, f, g)$,
  - $Q = \{1, 2\}$
  - $S_0 = \{1\}$
  - $I = \{. +\}$
  - $O = \{0, 1\}$
  - $f: Q \times I \to Q$
  - $g: Q \to O$

Figura 12.

Diagramas de las máquinas deterministas