# Funciones de obtención de datos

Depende del formato de los datos

Típicamente:

- Csv
- Json

Otros:

- Tsv
- Xml

# EJERCICIO 3

**Enunciado**:

1. Console.log() sirve para imprimir por consola. El objetivo es leer los distintos tipos de archivo provistos y conseguir imprimirlos por consola

# Tratamiento de datos

**Operaciones principales:**

- Convertir en número (enteros / flotantes)
- Convertir fechas

**Operaciones secundarias:**

- Ordenar datos

# Tratamiento de datos

## Función map:

```
const array1 = [1, 4, 9, 16];

// pass a function to map
const map1 = array1.map(x => x * 2);

console.log(map1);
// expected output: Array [2, 8, 18, 32]
```

```
const obj1 = [{name: “Mike”, age: 30},
{name: “Will”, age: 28}]

// pass a function to map
const map1 = obj1.map(x => x.name);

console.log(map1);
// expected output: Array [“Mike”, “Will”]
```


# Tratamiento de datos

## Operaciones principales:

- Convertir en número (enteros / flotantes)
- **Convertir fechas**

```
const formatDate = d3.timeParse("%d%m%Y");
```

- `%a` - abbreviated weekday name.
- `%A` - full weekday name.
- `%b` - abbreviated month name.
- `%B` - full month name.
- `%c` - date and time, as "%a %b %e %H:%M:%S %Y".
- `%d` - zero-padded day of the month as a decimal number [01,31].
- `%e` - space-padded day of the month as a decimal number [ 1,31]; equivalent to `%_d`.
- `%H` - hour (24-hour clock) as a decimal number [00,23].
- `%I` - hour (12-hour clock) as a decimal number [01,12].
- `%j` - day of the year as a decimal number [001,366].
- `%m` - month as a decimal number [01,12].
- `%M` - minute as a decimal number [00,59].
- `%L` - milliseconds as a decimal number [000, 999].


# Tratamiento de datos

- `%p` - either AM or PM.
- `%S` - second as a decimal number [00,61].
- `%U` - week number of the year (Sunday as the first day of the week) as a decimal number [00,53].
- `%w` - weekday as a decimal number [0(Sunday),6].
- `%W` - week number of the year (Monday as the first day of the week) as a decimal number [00,53].
- `%x` - date, as "%m/%d/%Y".
- `%X` - time, as "%H:%M:%S".
- `%y` - year without century as a decimal number [00,99].
- `%Y` - year with century as a decimal number.
- `%Z` - time zone offset, such as "-0700".
- `%%` - a literal "%" character.

# EJERCICIO 4

**Enunciado**:

1. Convertir los datos  pertinentes en enteros
2. Convertir los datos pertinentes en fechas

**Ayuda**:

```
const formatDate = d3.timeParse("%d%m%Y");
```

https://d3-wiki.readthedocs.io/zh_CN/master/Time-Formatting/

# Data binding (enlazando datos)

**Data / datum:** Sobre una selección, inyecta datos. Con data, une los datos (join)

**enter:**

- Compara datos con la selección y deja solo los que no tienen elementos en el DOM.
- Crea una selección “fantasma”, donde hay que hacer un append

**update** (no es una función)**:**

- Permite modificar los elementos existentes

**exit:**

- Compara datos con la selección y deja solo los elementos que no tienen datos (sobran).
- Permite eliminar los elementos que ya no tienen datos

# EJERCICIO 5

**Enunciado**:

1. Enter: Crear divs y etiquetas <p> para mostrar el contenido de datos.json

# EJERCICIO 6

**Enunciado**:

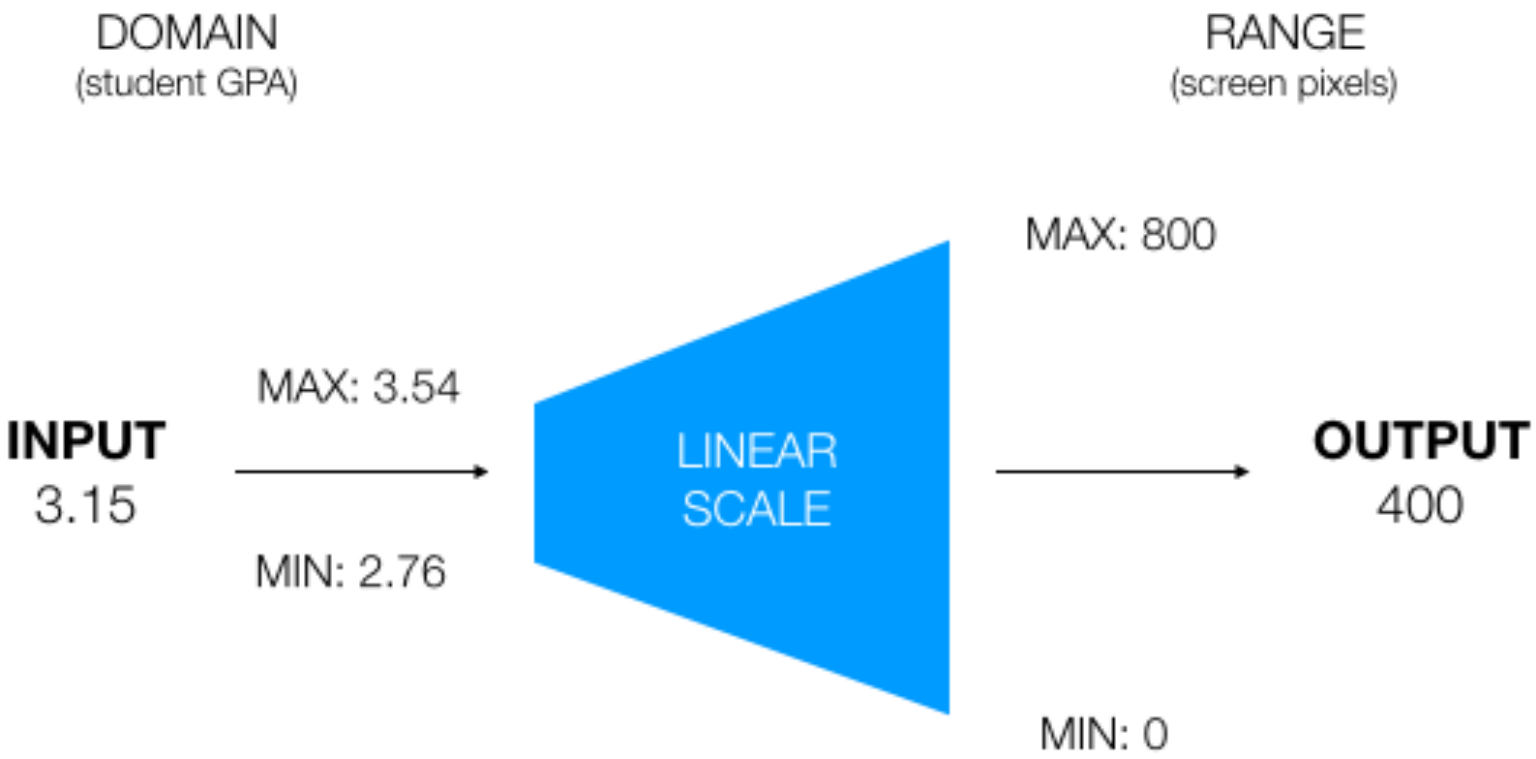
1. Update y exit: En el ejercicio anterior, crear un botón que filtre los países de Europa, y otro que filtre los países de Sudamérica (el resto ya ganarán algo algún día, suponemos).


109m
90m
23m
11m
12.5m
8.6m (tall)
20m
6m
11m
6m
8.5m
21m
27m
1.9m
38m
31m
20m (tall)
120m
150m

# Escala

# Escala




DOMAIN
(student GPA)
RANGE
(screen pixels)
MAX: 800
INPUT
3.15
MAX: 3.54
MIN: 2.76
LINEAR
SCALE
OUTPUT
400
MIN: 0

# Escala: funciones útiles


**d3.max(array):**

- Recibe un array de x posiciones y devuelve el valor máximo que encuentre en el array.

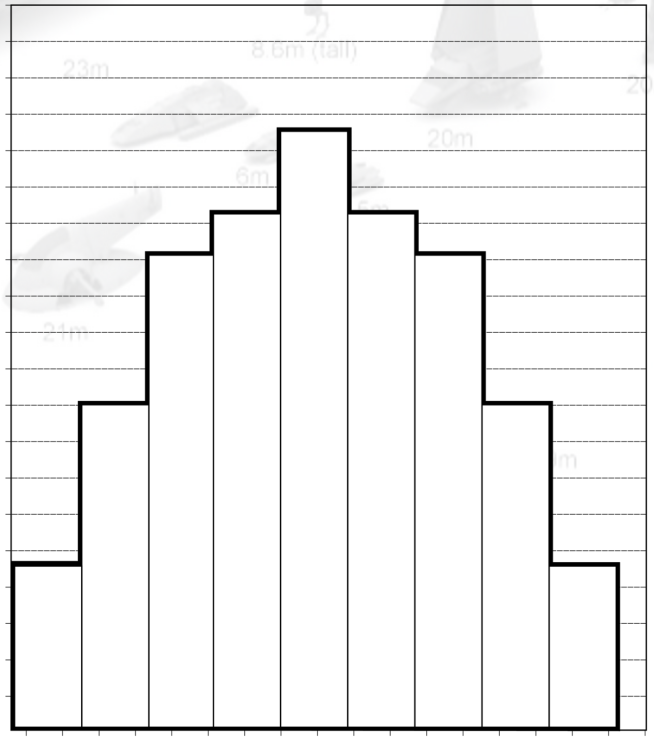
**d3.min(array):**

- Recibe un array de x posiciones y devuelve el valor mínimo que encuentre en el array.

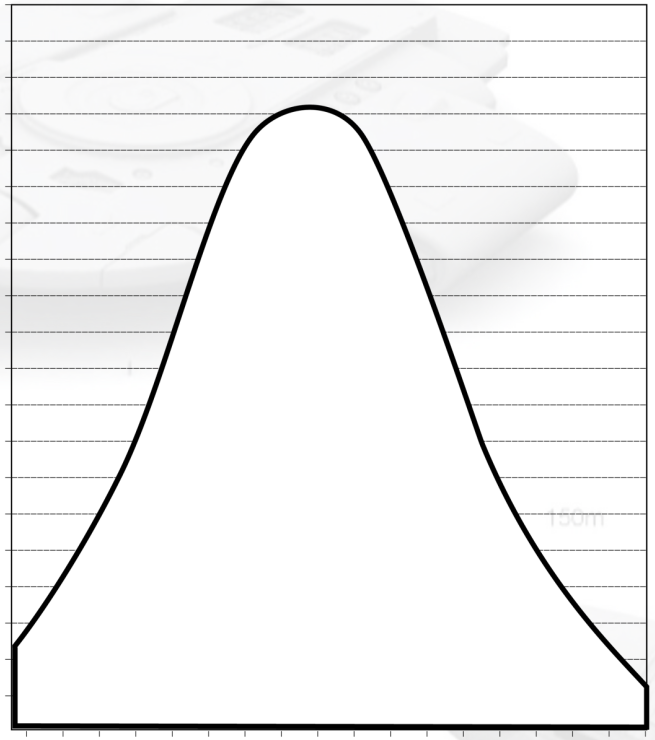
**d3.extent(array):**

- Recibe un array de x posiciones y devuelve un array de dos posiciones con el mínimo en la primera posición y el máximo en la segunda

# **Escala:** Datos discretos vs datos continuos

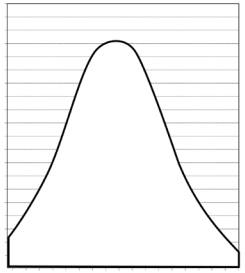


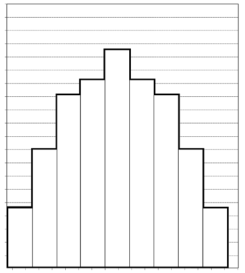
Datos discretos



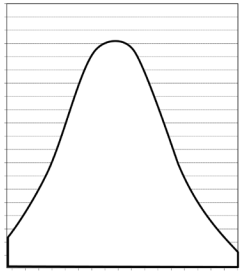
Datos continuos

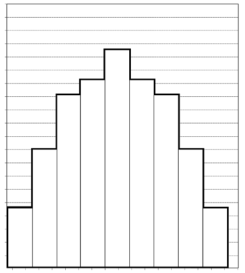
# Tipos de escala





Input





1. scaleLinear
2. scaleSqrt
3. scalePow
4. scaleLog
5. scaleTime
6. scaleSequential

1. scaleQuantize
2. scaleQuantile
3. scaleThreshold

1. scaleOrdinal
2. scaleBand
3. scalePoint

Output

# EJERCICIO 7

**Enunciado**:

1. Escalas: definir una función de escala lineal que permita averiguar el tamaño a 1/50 de la altura de cualquier edificio. La función recibirá un dato en metros y lo hará 50 veces más pequeño.

# EJERCICIO 8

**Enunciado**:
Datos.json contiene una serie de ciudades con la distancia desde cada una de ellas a Berlin, en km.

1. Escalas: representar en una línea horizontal la distancia en km que hay desde Berlin a las distintas ciudades. Da igual la geometría que usemos para representar dicha ciudad. Hay que incluir el nombre.