

Analysis of the Constitutional Council's decisions



HERRMANN
RENARD



METHODOLOGY

Loop over 342 pages

This Python loop iterates through the 342 pages of the static site belonging to the Constitutional Council. Each page contains 20 decisions.

Collection of decision details

Within this loop, various functions are employed to extract information pertaining to each decision :

- Year
- Type
- Solution
- Presiding judge
- Length

Dataframe creation

The extracted information is subsequently transformed into a pandas DataFrame named "constitdf," which is then written to a CSV file.

Graphical analysis

The transformed information, now in CSV format, is then analyzed using various graphs.





FOCUS I : CREATION OF THE DATAFRAME

1. Year, type and solution
2. Presiding judge
3. Length of decision



FR

Décision n° 2023-6278/6282
SEN du 15 mars 2024

SEN, Hauts-de-Seine, M. Philippe
PEMEZEC et autres

[Rejet]

Décision n° 2024-304 L du 14
mars 2024

Nature juridique de certaines
dispositions des articles L. 7551 du code
de l'éducation et L. 34111 du code de la
défense

[Législatif]

Décision n° 2023-
6272/6277/6280 SEN du 7 mars
2024

SEN, Moselle, M. Jean-Louis MASSON et
autres

[Rejet]

Décision n° 2023-1080 QPC du
6 mars 2024

Société de la Fontaine [Double degré de
juridiction pour l'examen d'un incident
contentieux relatif à l'exécution d'une

Elements >> 1 1 2

```
</article>
</div>
<div class="views-row"></div>
<div class="views-row">
  <article class="node node--type-decision node--promoted node--view-mode-search-result">
    <a href="/decision/2024/20236278_6282SEN.htm" rel="bookmark" title="Décision n°&nbsp;2023-6278/6282 SEN du 15 mars 2024" data-once="external-link">
      <div class="node__content"> flex
        <div class="left"></div>
        <div class="right">
          <div class="title">Décision n°&nbsp;2023-6278/6282 SEN du 15 mars 2024</div>
          <div class="field field--name-field-titre-complet field--type-string-long field--label-hidden field__item">SEN, Hauts-de-Seine, M.&nbsp;Philippe PEMEZEC et autres</div>
          <div class="field field--name-field-solution-normalisee field--type-string field--label-hidden field__item">== $0
            ::before
            "Rejet"
            ::after
          </div>
        </div>
      </div>
    </a>
  </article>
</div>
<div class="views-row"></div>
<div class="views-row"></div>
<div class="views-row"></div>
<div class="views-row"></div>
```

div.field.field--name-field-solution-normalisee.field--type-string

Styles Computed Layout Event Listeners >>

Filter :hov .cls

+ Code + Text All changes saved

Connect

for x in range(1, 343): # il faut aller jusqu'à la fin de la page

webpage = requests.get(url_prefix + str(x))

soup = BeautifulSoup(webpage.content, "html.parser")

aas = soup.find_all("div", class_="views-row")

for a in aas:

href = a.find("a").get("href")

sublist = [href]

title = a.find("div", class_="title").text.split(" ")

year = None

capitalized_word = None

for word in title:

year_match = re.search(r'\b(\d{4})\b', word)

if year_match:

year = year_match.group(1)

capitalized_word_match = re.search(r'\b([A-Z]{2,})\b', word)

if capitalized_word_match:

capitalized_word = capitalized_word_match.group(1)

if year:

sublist.append(year)

else:

sublist.append("Année pas trouvée")

if capitalized_word:

sublist.append(capitalized_word)

else:

sublist.append("L")


try:

solution = a.find("div", class_="field field--name-field-solution-normalisee field--type-string field__item")

except Exception as e:

solution="introuvable"

sublist.append(solution)



FR

M. Philippe FEMEZEC et de
Mme Catherine CANDELIER,
M. Gilles MERGY et M. David
MAUGER sont rejetées.

Article 2. - Cette décision sera publiée
au Journal officiel de la République
française et notifiée dans les
conditions prévues à l'article 18 du
règlement applicable à la procédure
suivie devant le Conseil
constitutionnel pour le contentieux
de l'élection des députés et des
sénateurs.

Jugé par le Conseil constitutionnel
dans sa séance du 14 mars 2024, où
siégeaient : M. Laurent FABIOUS,
Président, Mme Jacqueline
GOURAULT, M. Alain JUPPÉ, Mmes
Corinne LUQUIENS, Véronique
MALBEC, MM. Jacques MÉZARD,
Michel PINAULT et François
SÉNERS.

Rendu public le 15 mars 2024.

JORF n°0066 du 19 mars 2024, texte
n° 41
ECLI : FR : CC : 2024 : 2023.6278.SEN

Elements

<p class="considerant">...</p>

<p class="considerant">...</p>

<p class="considerant">...</p>

<blockquote>

<p>...</p>

<p> == \$0

"Jugé par le Conseil
constitutionnel dans sa séance
du 14 mars 2024, où
siégeaient : :
M. Laurent FABIOUS,
Président, Mme Jacqueline
GOURAULT, M. Alain JUPPÉ, Mmes
Corinne LUQUIENS, Véronique
MALBEC, MM. Jacques MÉZARD,
Michel PINAULT et François
SÉNERS."

" ";

" Rendu public le 15 mars
2024."

" ";
</p>

<p>...</p>

</blockquote>

::after

</div>

</div>

<div class="cartouche-decision-print">
...</div>

<div class="wrapper-dernieres-decision
s">...</div>

<a href="/les-decisions" class="lien-t
outes-decisions" data-once="external-l
ink">Toutes les décisions

</div>

</article>

</div>

</div>

</div>

</main>

type-text-long.field--label-hidden.field__item

blockquote

p

Styles

Computed

Layout

Event Listeners

>>

Filter

:hov .cls

+

+ Code + Text All changes saved

Connect

from collections import defaultdict

import pandas as pd

import time

import matplotlib.pyplot as plt

def extract_judge_name(decision_url):

response = requests.get(decision_url)

if response.status_code == 200:

soup = BeautifulSoup(response.content, 'html.parser')

blockquote_tag = soup.find('blockquote').text.strip()

match = re.search(r'par le Conseil constitutionnel.*?([A-Z]{4,})',

if match:

judge_name = match.group(1)

if judge_name == "DEBR":

judge_name += "É"

return judge_name

else:

return "Le nom du juge n'a pas été mentionné dans la décision.

else:

print("La décision n'a pas pu être récupérée")

return None

url_prefix = "https://www.conseil-constitutionnel.fr/les-decisions?cc-page

main_list = []

for x in range(1, 343): # il faut aller jusqu'à 343 et pas 342 apparemment

webpage = requests.get(url_prefix + str(x))

soup = BeautifulSoup(webpage.content, "html.parser")

aas = soup.find_all("div", class_="views-row")

for a in aas:

href = a.find("a").get("href")

sublist = [href]

title = a.find("div", class_="title").text.split(" ")

year = None

capitalized_word = None

for word in title:



FR



DÉCISION

Décision n° 2023-6278/6282 SEN du 15 mars 2024

SEN, Hauts-de-Seine, M. Philippe

PEMEZEC et autres

[Rejet]

Décision n° 2023-6278/6282 SEN



Version PDF de la décision

Pdf 162.51 Ko



Lien stable de la décision

LE CONSEIL CONSTITUTIONNEL A ÉTÉ SAISI le 29 septembre 2023 d'une requête présentée pour M. Philippe PEMEZEC, candidat à l'élection sénatoriale qui s'est déroulée dans les Hauts-de-Seine, par Me Bernard Cazin, avocat au barreau de Paris, tendant à l'annulation des opérations électorales auxquelles il a été procédé dans ce

Elements >> 2 1

off-canvas-main-canvas>
▶ <header role="banner" class="sticky">...</header>
▶ <div class="region region-breadcrumb">...</div>
▶ <div class="region region-highlighted">...</div>
▼ <main role="main">

 ▼ <div class="layout-content ">
 ▼ <div class="region region-content">
 ▼ <div id="block-cc-content" class="block block-system block-system-main-block">
 ▼ <article class="node node--type-decision node--promoted node--view-mode-full">
 ▼ <div class="node__content"> flex
 ▶ <header>...</header> flex
 ▶ <aside class="sidebar">...</aside>
 ▼ <div class="wrapper-content">
 ▼ <div class="clearfix text-formatted field field--name-field-contenu-original field--type-text-long field--label-hidden field__item">
...
▼ <p> == \$0
"LE CONSEIL CONSTITUTIONNEL A ÉTÉ SAISI le 29 septembre 2023 d'une requête présentée pour M. Philippe PEMEZEC, candidat à l'élection sénatoriale qui s'est déroulée dans les Hauts-de-Seine, par Me Bernard Cazin, avocat au barreau de Paris, tendant à l'annulation des opérations électorales auxquelles il a été procédé dans ce département, le 24 septembre 2023, en vue de la désignation de sept sénateurs. Elle a été enregistrée au secrétariat général du Conseil constitutionnel sous le n° 2023-6278 SEN."

" Il a également été saisi le 3 octobre 2023 d'une requête tendant aux mêmes fins.

original.field--type-text-long.field--label-hidden.field__item p

Styles Computed Layout Event Listeners >>

Filter :hov .cls +

+ Code + Text All changes saved Connect

capitalized_word = None

for word in title:
 year_match = re.search(r'\b(\d{4})\b', word)
 if year_match:
 year = year_match.group(1)

 capitalized_word_match = re.search(r'\b([A-Z]{2,})\b', word)
 if capitalized_word_match:
 capitalized_word = capitalized_word_match.group(1)

if year:
 sublist.append(year)
else:
 sublist.append("Année pas trouvée")

if capitalized_word:
 sublist.append(capitalized_word)
else:
 sublist.append("L")

try:
 solution = a.find("div", class_="field field--name-field-solution-
except Exception as e:
 solution="introuvable"
sublist.append(solution)

detail_webpage = requests.get("https://www.conseil-constitutionnel.fr"
detail_soup = BeautifulSoup(detail_webpage.content, "html.parser")
detail_div = detail_soup.find("div", class_="clearfix text-formatted fi

if detail_div:
 text = detail_div.getText()
 debut = re.search("LE CONSEIL CONSTITUTIONNEL A ÉTÉ SAISI", text)
 length = len(text)
 sublist.append(length)





Suivre l'activité du conseil constitutionnel

```
:hov .cls +
```

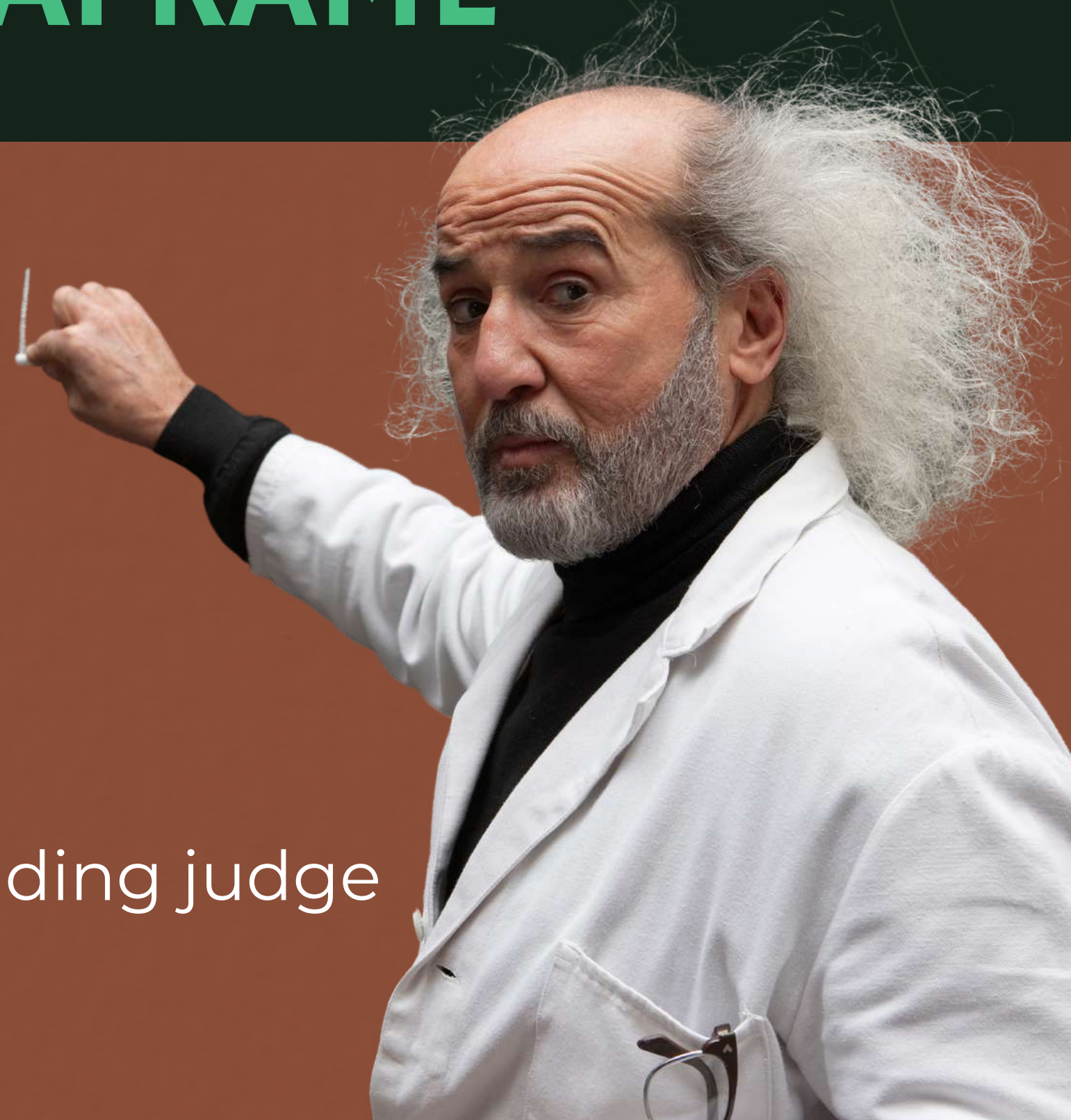
```
if capitalized_word:
    sublist.append(capitalized_word)
else:
```




FOCUS 2 :

ANALYSIS OF THE DATAFRAME

1. Dataframe
2. Number of decisions per year
3. Average length of decisions per year
4. Number of decisions by type since 1958
5. Number of decisions issued by each presiding judge




```
constitdf = pd.DataFrame(main_list, columns=["URL", "Année", "Type de décision", "Solution", "Length", "Juge"])
constitdf.to_csv("constitdf.csv", encoding="utf8")
print(constitdf)
```

```
[580 rows x 6 columns]
```



		URL	Année	Type de décision	\
0	/decision/2024/20231081QPC.htm		2024	QPC	
1	/decision/2024/20231082QPC.htm		2024	QPC	
2	/decision/2024/20236278_6282SEN.htm		2024	SEN	
3	/decision/2024/2024304L.htm		2024	L	
4	/decision/2024/20236272_6277_6280SEN.htm		2024	SEN	
..		
575	/decision/2022/20223RIP.htm		2022	RIP	
576	/decision/2022/20221017_1018QPC.htm		2022	QPC	
577	/decision/2022/20221015QPC.htm		2022	QPC	
578	/decision/2022/20221016QPC.htm		2022	QPC	
579	/decision/2022/20221014QPC.htm		2022	QPC	

	Solution	Length	Juge
0	Conformité – réserve	9444	FABIUS
1	Non lieu à statuer	8176	FABIUS
2	Rejet	12085	FABIUS
3	Législatif	3620	FABIUS
4	Rejet	5302	FABIUS
..
575	Non conformité	5756	FABIUS
576	Conformité	7240	FABIUS
577	Conformité	24222	FABIUS
578	Conformité	10730	FABIUS
579	Conformité	9557	FABIUS

[580 rows x 6 columns]



+ Code + Texte

Connecter



```
plt.title('Number of decisions issued by each presiding judge')
plt.xlabel('Name of the presiding judges')
plt.ylabel('Number of decisions')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
#3rd graph graphique: number of decisions per year : linear graph
```

```
# Count the number of decisions per year
```

```
decisions_per_year = constitdf['Année'].value_counts().sort_index()
```

```
# Plot linear graph for number of decisions per year
```

```
plt.figure(figsize=(15, 6))
```

```
plt.plot(decisions_per_year.index, decisions_per_year, marker='o', color='red', linestyle='-')
```

```
plt.xlabel('Year')
```

```
plt.ylabel('Number of decisions')
```

```
plt.title('Number of decisions per year')
```

```
plt.xticks(rotation=45) # Rotate x-axis labels for better readability
```

```
plt.grid(True) # Add gridlines for better readability
```

```
plt.xlim(min(decisions_per_year.index), max(decisions_per_year.index)) # Set x-axis limits
```

```
plt.xticks(rotation='vertical') # Rotate x-axis labels vertically
```

```
plt.tight_layout()
```

```
plt.show()
```

```
#4th graph: number of decision for each type of decision since 1958: histogram graph
```

```
# Count the number of decisions for each type of capitalized word:
```

```
type_decision_counts = constitdf['Type de décision'].value_counts()
```

```
# Plot bar graph
```

```
plt.figure(figsize=(15, 6))
```

```
type_decision_counts.plot(kind='bar', color='pink')
```

```
plt.xlabel('Type of decisions')
```

```
plt.ylabel('Number of decisions')
```

```
plt.title('Number of decisions by type since 1958')
```

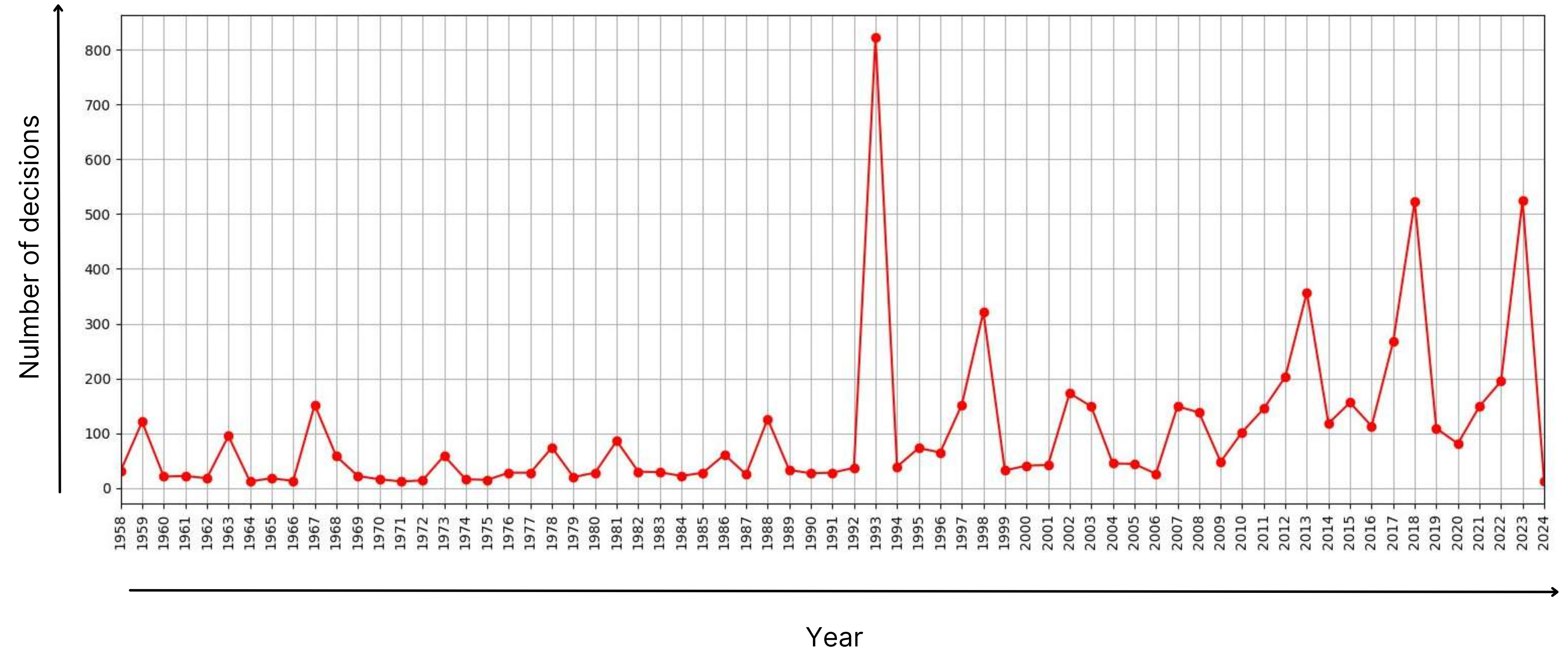
```
plt.xticks(rotation=45)
```

```
plt.tight_layout()
```

```
plt.show()
```



Number of decisions per year





+ Code + Texte

Connecter



```
elif 280 <= x <= 288 and judge_name == "ECLI":
    judge_name = "Unidentified"
elif judge_name == "FRET": # Misspelled last name
    judge_name = "FREY"
elif judge_name == "GUENA": # The site is very inconsistent in the way they write SURNAMES
    judge_name = "GUÉNA"
sublist.append(judge_name) # we put the judges' names in our sublist

main_list.append(sublist)# we put all the sublists into our main list in order to be able to create a dataframe
```

```
constitdf = pd.DataFrame(main_list, columns=["URL", "Année", "Type de décision", "Solution", "Length", "Juge"]) # we create a dataframe
constitdf.to_csv("constitdf.csv", encoding="utf8") # we save the data to CSV
```

```
#1st graph: mean decisions' length per year: linear graph
```

```
# Convert 'Length' column to numeric
```

```
constitdf['Length'] = pd.to_numeric(constitdf['Length'], errors='coerce')
```

```
# Calculate mean decisions' length per year
```

```
mean_length_per_year = constitdf.groupby('Année')['Length'].mean()
```

```
# Plot linear graph for mean decisions' length per year
```

```
plt.figure(figsize=(15, 6))
```

```
plt.plot(mean_length_per_year.index, mean_length_per_year, marker='o', color='green', linestyle='-')
```

```
plt.xlabel('Year')
```

```
plt.ylabel('Average length of decisions')
```

```
plt.title('Average length of decisions per year')
```

```
plt.grid(True)
```

```
plt.xlim(min(mean_length_per_year.index), max(mean_length_per_year.index)) # Set x-axis limits
```

```
plt.xticks(rotation='vertical') # Rotate x-axis labels vertically
```

```
plt.tight_layout()
```

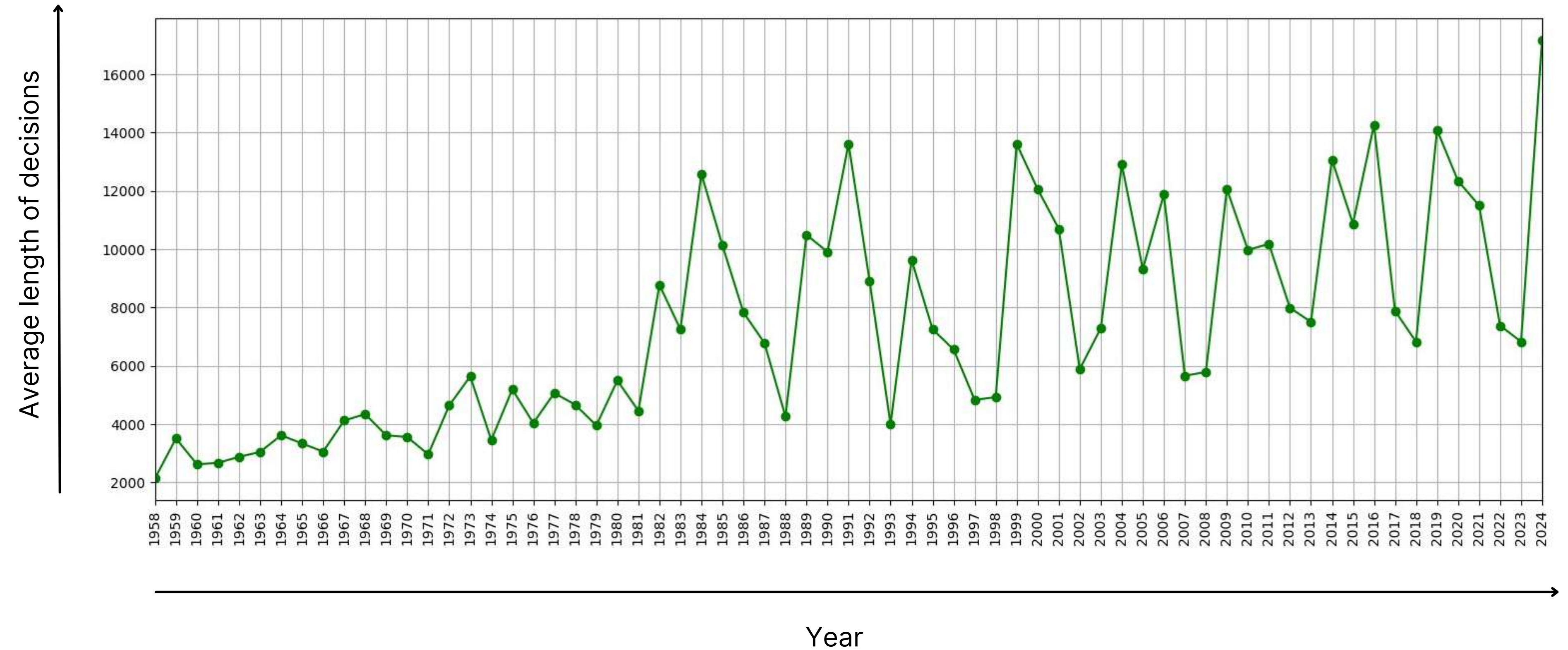
```
plt.show()
```

```
#2nd graph: number of decisions taken by each President of the Conseil Constitutionnel: histogram graph
```

```
# Group data by judge and count the number of decisions for each President
```

```
judge_counts = constitdf['Juge'].value_counts()
```


Average length of decisions per year





+ Code + Texte

Connecter

```
#3rd graph graphique: number of decisions per year : linear graph

# Count the number of decisions per year
decisions_per_year = constitdf['Année'].value_counts().sort_index()

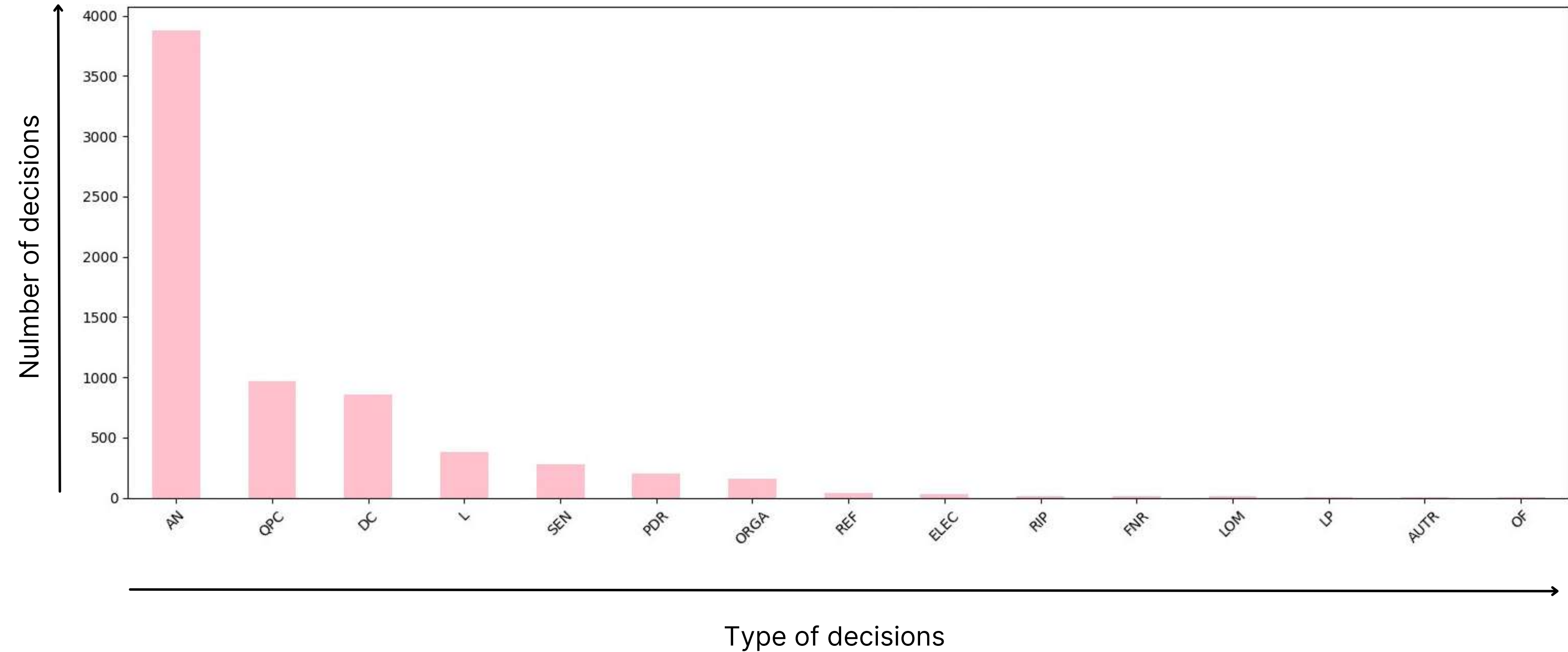
# Plot linear graph for number of decisions per year
plt.figure(figsize=(15, 6))
plt.plot(decisions_per_year.index, decisions_per_year, marker='o', color='red', linestyle='-')
plt.xlabel('Year')
plt.ylabel('Number of decisions')
plt.title('Number of decisions per year')
plt.xticks(rotation=45) # Rotate x-axis labels for better readability
plt.grid(True) # Add gridlines for better readability
plt.xlim(min(decisions_per_year.index), max(decisions_per_year.index)) # Set x-axis limits
plt.xticks(rotation='vertical') # Rotate x-axis labels vertically
plt.tight_layout()
plt.show()
```

```
#4th graph: number of decision for each type of decision since 1958: histogram graph
```

```
# Count the number of decisions for each type of capitalized word:
type_decision_counts = constitdf['Type de décision'].value_counts()
```

```
# Plot bar graph
plt.figure(figsize=(15, 6))
type_decision_counts.plot(kind='bar', color='pink')
plt.xlabel('Type of decisions')
plt.ylabel('Number of decisions')
plt.title('Number of decisions by type since 1958')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```


Number of decisions by type since 1958





+ Code + Texte

Connecter

```
mean_length_per_year = constitdf.groupby('Année')['Length'].mean()

# Plot linear graph for mean decisions' length per year
plt.figure(figsize=(15, 6))
plt.plot(mean_length_per_year.index, mean_length_per_year, marker='o', color='green', linestyle='-')
plt.xlabel('Year')
plt.ylabel('Average length of decisions')
plt.title('Average length of decisions per year')
plt.grid(True)
plt.xlim(min(mean_length_per_year.index), max(mean_length_per_year.index)) # Set x-axis limits
plt.xticks(rotation='vertical') # Rotate x-axis labels vertically
plt.tight_layout()
plt.show()
```

```
#2nd graph: number of decisions taken by each President of the Conseil Constitutionnel: histogram graph
```

```
# Group data by judge and count the number of decisions for each President
judge_counts = constitdf['Juge'].value_counts()
```

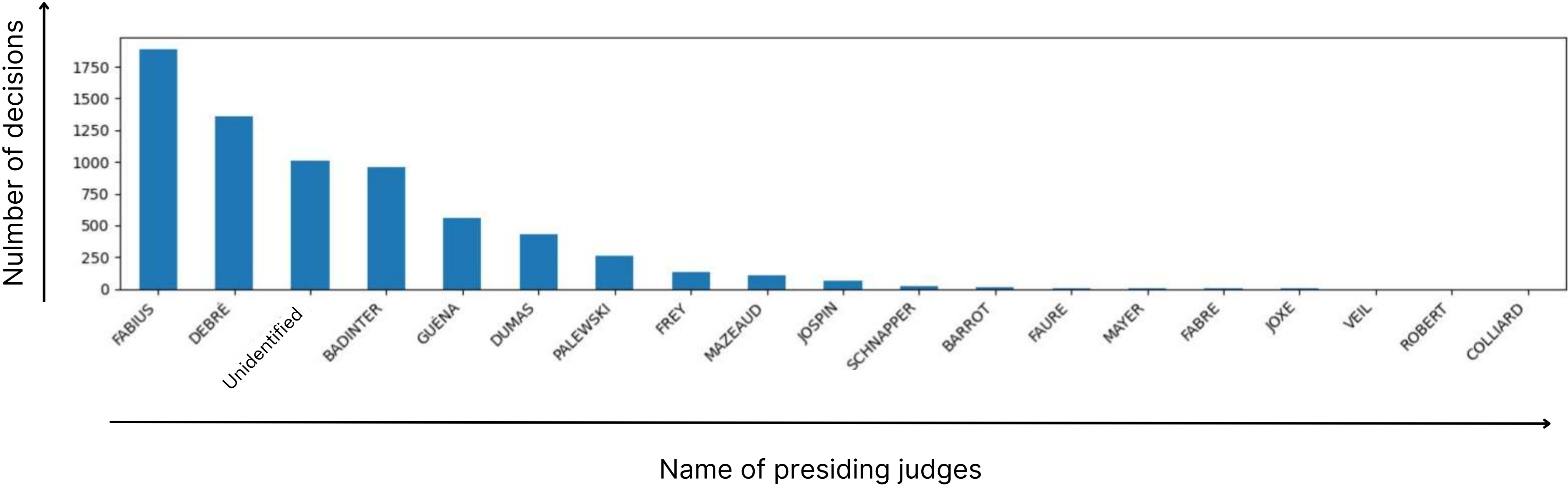
```
# Plot histogram
plt.figure(figsize=(15, 6))
judge_counts.plot(kind='bar')
plt.title('Number of decisions issued by each presiding judge')
plt.xlabel('Name of the presiding judges')
plt.ylabel('Number of decisions')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
#3rd graph graphique: number of decisions per year : linear graph
```

```
# Count the number of decisions per year
decisions_per_year = constitdf['Année'].value_counts().sort_index()
```

```
# Plot linear graph for number of decisions per year
plt.figure(figsize=(15, 6))
plt.plot(decisions_per_year.index, decisions_per_year, marker='o', color='red', linestyle='-')
plt.xlabel('Year')
plt.ylabel('Number of decisions')
```


Number of decisions issued by each presiding judge





CONCLUSION FINDINGS





THANK YOU ! QUESTIONS ?



**Jean-Baptiste
HERRMANN**



**Pénélope
RENARD**