```python
import pandas as pd
import numpy as np


# VALORI MANCANTI


# In python i valori mancanti sono indicati con NaN. Possiamo gestirli in 3 modi
# - Tenerli
# - Rimuoverli
# - Sostituirli

# Si noti che non c'è un approccio corretto al 100%; In base alla situazione si decide quale approccio utilizzare.
# Ciascuno di questi approcci ha pro e contro. Vediamoli in dettaglio

# - TENERE I NULL => PRO: E' la cosa più semplice da fare e non manipola i dati true. CONTRO: Molti metodi non funzionano quando si ha a che fare con i null
# - RIMOZIONE DATI NaN ==> PRO: E' facile da usare CONTRO: E' possibile perdere la maggior parte dei dati e/o perdere informazioni utili.


# Per studiare i valori NaN utilizzeremo il dataframe movie_scores:


movie_scores = pd.read_csv("movie_scores.csv")
movie_scores
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
1	NaN	NaN	NaN	NaN	NaN	NaN
2	Hugh	Jackman	51.0	m	NaN	NaN
3	Oprah	Winfrey	66.0	f	6.0	8.0
4	Emma	Stone	31.0	f	7.0	9.0
5	NaN	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN	NaN
7	Emma	Stone	31.0	f	7.0	9.0
8	Kiefer	sutherland	60.0	M	6.0	12.0

```python
# Utilizzo di isnull per vedere quali valori sono nulli


movie_scores.isnull()
```

```
   first_name  last_name    age    sex  pre_movie_score
post_movie_score
0       False      False  False  False            False
False
1        True       True   True   True             True
True
2       False      False  False  False             True
True
3       False      False  False  False            False
False
4       False      False  False  False            False
False
5        True       True   True   True             True
True
6        True       True   True   True             True
True
7       False      False  False  False            False
False
8       False      False  False  False            False
False


# In corrispondenza di True avremo valori nulli, in corrispondenza di
False avremo valori non nulli


# Utilizzo di notnull per vedere quali valori NON sono nulli. Si
ragiona in modo opposto a isnull


movie_scores.notnull()
```

```
   first_name  last_name    age    sex  pre_movie_score
post_movie_score
0        True       True   True   True             True
True
1       False      False  False  False            False
False
2        True       True   True   True            False
False
3        True       True   True   True             True
True
4        True       True   True   True             True
True
5       False      False  False  False            False
False
6       False      False  False  False            False
False
```

```
7        True       True   True   True              True
True
8        True       True   True   True              True
True
```

```
# Abbiamo False in corrispondenza dei valori nulli.
```

```
# Possiamo anche lavorare su singola colonna. Prendiamo,
# a titolo di esempio, la colonna age. Se vogliamo vedere quali valori
sono
# NON nulli, è sufficiente applicare la funzione notnull alla colonna
considerata
```

```
movie_scores["age"].notnull()
```

```
0     True
1    False
2     True
3     True
4     True
5    False
6    False
7     True
8     True
Name: age, dtype: bool
```

```
# E' quindi null solo il valore di age corrispondente all'indice 1.
```

```
# Ovviamente possiamo filtrare in base alla funzione notnull.
Supponiamo
# di voler estrarre le righe del nostro Df in corrispondenza delle
quali il valore
# age è NON nullo. Allora:
```

```
mask = movie_scores["age"].notnull()
movie_scores[mask]
```

```
  first_name   last_name   age sex  pre_movie_score  post_movie_score
0        Tom       Hanks  63.0   m              8.0              10.0
2       Hugh     Jackman  51.0   m              NaN               NaN
3      Oprah     Winfrey  66.0   f              6.0               8.0
```

```
4       Emma       Stone  31.0   f              7.0               9.0
7       Emma       Stone  31.0   f              7.0               9.0
8     Kiefer  sutherland  60.0   M              6.0              12.0
```

```
# La riga con indice 1 non è presente.  CVD
```

```
movie_scores
```

```
  first_name   last_name   age  sex  pre_movie_score  post_movie_score
0        Tom       Hanks  63.0    m              8.0              10.0
1        NaN         NaN   NaN  NaN              NaN               NaN
2       Hugh     Jackman  51.0    m              NaN               NaN
3      Oprah     Winfrey  66.0    f              6.0               8.0
4       Emma       Stone  31.0    f              7.0               9.0
5        NaN         NaN   NaN  NaN              NaN               NaN
6        NaN         NaN   NaN  NaN              NaN               NaN
7       Emma       Stone  31.0    f              7.0               9.0
8     Kiefer  sutherland  60.0    M              6.0              12.0
```

```
# Possiamo anche ragionare al contrario con la funzione isnull.
Supponiamo di voler
# considerare le righe del nostro Df in corrispondenza delle quali il
valore
# age è nullo Allora:
```

```
mask = movie_scores["age"].isnull()
mask
```

```
0    False
1     True
2    False
3    False
4    False
5     True
6     True
7    False
8    False
Name: age, dtype: bool
```

```
movie_scores[mask]
```

```
  first_name last_name  age  sex  pre_movie_score  post_movie_score
1        NaN       NaN  NaN  NaN              NaN               NaN
```

```
5        NaN       NaN  NaN  NaN              NaN               NaN
6        NaN       NaN  NaN  NaN              NaN               NaN
```

```
# CVD
```

```
# Possiamo applicare anche un filtro con più condizioni.

# Supponiamo di voler estrarre le colonne del Df in corrispondenza
delle quali
# il valore di pre_movie_score è null ed il valore di sex è diverso da
null. Allore:
```

```
mask = (movie_scores["pre_movie_score"].isnull()) &
(movie_scores["sex"].notnull())
mask
```

```
0    False
1    False
2     True
3    False
4    False
5    False
6    False
7    False
8    False
dtype: bool
```

```
movie_scores[mask]
```

```
  first_name last_name   age sex  pre_movie_score  post_movie_score
2       Hugh   Jackman  51.0   m              NaN               NaN
```

```
# CVD
```

```
# ABBIAMO FINORA STUDIATO I METODI null, notnull E LI ABBIAMO MESSI
NEL mask. VEDIAMO ORA
# COME COMPORTARCI DI FRONTE A VALORI MANCANTI. SI RICORDI CHE CI SONO
3 COMPORTAMENTI DIVERSI
# CHE POSSIAMO ADOTTARE:
# - NON FARE NULLA
```

```
# - TOGLIERE I VALORI MANCANTI
# -
```

```
# TENERE I DATI MANCANTI
```

```
movie_scores
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
1	NaN	NaN	NaN	NaN	NaN	NaN
2	Hugh	Jackman	51.0	m	NaN	NaN
3	Oprah	Winfrey	66.0	f	6.0	8.0
4	Emma	Stone	31.0	f	7.0	9.0
5	NaN	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN	NaN
7	Emma	Stone	31.0	f	7.0	9.0
8	Kiefer	sutherland	60.0	M	6.0	12.0

```
# FINE
```

```
# ELIMINARE I DATI MANCANTI (dropna)
```

```
movie_scores.dropna()  # Elimina tutte quelle righe che presentano
almeno un valore NULL
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
3	Oprah	Winfrey	66.0	f	6.0	8.0
4	Emma	Stone	31.0	f	7.0	9.0
7	Emma	Stone	31.0	f	7.0	9.0
8	Kiefer	sutherland	60.0	M	6.0	12.0

```
movie_scores
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
1	NaN	NaN	NaN	NaN	NaN	NaN
2	Hugh	Jackman	51.0	m	NaN	NaN

```
3      Oprah     Winfrey  66.0    f              6.0               8.0
4       Emma       Stone  31.0    f              7.0               9.0
5        NaN         NaN   NaN  NaN              NaN               NaN
6        NaN         NaN   NaN  NaN              NaN               NaN
7       Emma       Stone  31.0    f              7.0               9.0
8     Kiefer  sutherland  60.0    M              6.0              12.0
```

```
# Possiamo anche eliminare le sole righe nelle quali un almeno un
certo numero di valori è mancante. Per farlo, inseriamo il parametro
# thresh all'interno della nostra funzione dropna


# Eliminazione delle righe che hanno almeno un valore NON nullo

movie_scores.dropna(thresh=1)
```

```
  first_name   last_name   age sex  pre_movie_score  post_movie_score
0        Tom       Hanks  63.0   m              8.0              10.0
2       Hugh     Jackman  51.0   m              NaN               NaN
3      Oprah     Winfrey  66.0   f              6.0               8.0
4       Emma       Stone  31.0   f              7.0               9.0
7       Emma       Stone  31.0   f              7.0               9.0
8     Kiefer  sutherland  60.0   M              6.0              12.0
```

```
movie_scores.dropna(thresh=2)
```

```
  first_name   last_name   age sex  pre_movie_score  post_movie_score
0        Tom       Hanks  63.0   m              8.0              10.0
2       Hugh     Jackman  51.0   m              NaN               NaN
3      Oprah     Winfrey  66.0   f              6.0               8.0
4       Emma       Stone  31.0   f              7.0               9.0
7       Emma       Stone  31.0   f              7.0               9.0
8     Kiefer  sutherland  60.0   M              6.0              12.0
```

```
# Mantiene le righe che hanno almeno tre valori NON nulli

movie_scores.dropna(thresh=3)
```

```
  first_name   last_name   age sex  pre_movie_score  post_movie_score
0        Tom       Hanks  63.0   m              8.0              10.0
2       Hugh     Jackman  51.0   m              NaN               NaN
3      Oprah     Winfrey  66.0   f              6.0               8.0
```

```
4       Emma       Stone  31.0   f              7.0               9.0
7       Emma       Stone  31.0   f              7.0               9.0
8     Kiefer  sutherland  60.0   M              6.0              12.0
```

```
# Mantiene le righe che hanno almeno quattro valori NON nulli

movie_scores.dropna(thresh=4)
```

```
  first_name   last_name   age sex  pre_movie_score  post_movie_score
0        Tom       Hanks  63.0   m              8.0              10.0
2       Hugh     Jackman  51.0   m              NaN               NaN
3      Oprah     Winfrey  66.0   f              6.0               8.0
4       Emma       Stone  31.0   f              7.0               9.0
7       Emma       Stone  31.0   f              7.0               9.0
8     Kiefer  sutherland  60.0   M              6.0              12.0
```

```
# Mantiene le righe che hanno almeno cinque valori NON nulli (in
questo caso la riga 2 sparirà, perché
# I valori NON nulli sono 4, e 4<5



movie_scores.dropna(thresh=5)
```

```
  first_name   last_name   age sex  pre_movie_score  post_movie_score
0        Tom       Hanks  63.0   m              8.0              10.0
3      Oprah     Winfrey  66.0   f              6.0               8.0
4       Emma       Stone  31.0   f              7.0               9.0
7       Emma       Stone  31.0   f              7.0               9.0
8     Kiefer  sutherland  60.0   M              6.0              12.0
```

```
# POSSIAMO ANCHE LAVORARE PER COLONNE. FINORA, INFATTI,
IMPLICITAMENTE, ABBIAMO AVUTO axis=0. Cosa succede se ponessimo
# axis = 1?



movie_scores
```

```
  first_name   last_name   age  sex  pre_movie_score  post_movie_score
0        Tom       Hanks  63.0    m              8.0              10.0
1        NaN         NaN   NaN  NaN              NaN               NaN
2       Hugh     Jackman  51.0    m              NaN               NaN
3      Oprah     Winfrey  66.0    f              6.0               8.0
4       Emma       Stone  31.0    f              7.0               9.0
5        NaN         NaN   NaN  NaN              NaN               NaN
6        NaN         NaN   NaN  NaN              NaN               NaN
7       Emma       Stone  31.0    f              7.0               9.0
8     Kiefer  sutherland  60.0    M              6.0              12.0
```

```
movie_scores.dropna(axis=1)
```

```
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4, 5, 6, 7, 8]
```

```
# Non è stata selezionata alcuna colonna, questo perché TUTTE le
colonne presentanto ALMENO un valore
# pari a NULL
```

```
# Possiamo anche eliminare le righe in corrispondenza delle quali le
colonne corrispondenti presentano almeno
# un valore NON nullo. Per farlo, si inserisce il parametro subset
all'interno della nostra funzione fillna:
```

```
movie_scores
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
1	NaN	NaN	NaN	NaN	NaN	NaN
2	Hugh	Jackman	51.0	m	NaN	NaN
3	Oprah	Winfrey	66.0	f	6.0	8.0
4	Emma	Stone	31.0	f	7.0	9.0
5	NaN	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN	NaN
7	Emma	Stone	31.0	f	7.0	9.0
8	Kiefer	sutherland	60.0	M	6.0	12.0

```
movie_scores.dropna(subset=["sex"])
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
2	Hugh	Jackman	51.0	m	NaN	NaN
3	Oprah	Winfrey	66.0	f	6.0	8.0
4	Emma	Stone	31.0	f	7.0	9.0
7	Emma	Stone	31.0	f	7.0	9.0
8	Kiefer	sutherland	60.0	M	6.0	12.0

```
movie_scores.dropna(subset=["age","pre_movie_score"])
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
3	Oprah	Winfrey	66.0	f	6.0	8.0
4	Emma	Stone	31.0	f	7.0	9.0
7	Emma	Stone	31.0	f	7.0	9.0
8	Kiefer	sutherland	60.0	M	6.0	12.0

```
# PER CAPIRE MEGLIO dropna CON IL SUBSET CONSIDERIAMO IL NOSTRO Df
PRIVO DELLE RIGHE CHE PRESENTANO TUTTI
# VALORI null
```

```
movie_scores2 = pd.read_csv("movie_scores - Copia.csv")
movie_scores2
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
1	Hugh	Jackman	51.0	m	NaN	NaN
2	Oprah	Winfrey	66.0	f	6.0	8.0
3	Emma	Stone	31.0	f	7.0	9.0
4	Emma	Stone	NaN	f	7.0	9.0
5	Kiefer	sutherland	60.0	M	6.0	12.0

```
movie_scores2.dropna(subset=["first_name","last_name"])
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
1	Hugh	Jackman	51.0	m	NaN	NaN
2	Oprah	Winfrey	66.0	f	6.0	8.0
3	Emma	Stone	31.0	f	7.0	9.0
4	Emma	Stone	NaN	f	7.0	9.0
5	Kiefer	sutherland	60.0	M	6.0	12.0

```
# Non è stata eliminata alcuna riga, perché first_name e last_name non
presentano almeno un valore NULLO
```

```
# Invece, proviamo ora ad inserire una colonna senza valori nulli e
una colonna con valori nulli
```

```
movie_scores2.dropna(subset=["first_name","pre_movie_score"])
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
2	Oprah	Winfrey	66.0	f	6.0	8.0
3	Emma	Stone	31.0	f	7.0	9.0
4	Emma	Stone	NaN	f	7.0	9.0
5	Kiefer	sutherland	60.0	M	6.0	12.0

```
# La riga 1 è stata eliminata perché il valore corrispondente a
pre_movie_score è null


# FILLNA


# Con fillna possiamo avere a che fare con i valori mancanti.


# SOSTITUZIONE VALORI MANCANTI


movie_scores.fillna("SONO NULLO!!")
```

```
     first_name     last_name            age            sex
pre_movie_score  \
0           Tom         Hanks           63.0              m
8.0
1  SONO NULLO!!  SONO NULLO!!  SONO NULLO!!  SONO NULLO!!    SONO
NULLO!!
2          Hugh       Jackman           51.0              m    SONO
NULLO!!
3         Oprah       Winfrey           66.0              f
6.0
4          Emma         Stone           31.0              f
7.0
5  SONO NULLO!!  SONO NULLO!!  SONO NULLO!!  SONO NULLO!!    SONO
NULLO!!
6  SONO NULLO!!  SONO NULLO!!  SONO NULLO!!  SONO NULLO!!    SONO
NULLO!!
7          Emma         Stone           31.0              f
7.0
8        Kiefer    sutherland           60.0              M
6.0

  post_movie_score
0             10.0
1     SONO NULLO!!
2     SONO NULLO!!
3              8.0
4              9.0
5     SONO NULLO!!
6     SONO NULLO!!
7              9.0
8             12.0
```

```
# POSSIAMO SOSTITUIRE I VALORI MANCANTI ANCHE PER UNA SOLA COLONNA


movie_scores["pre_movie_score"]=
movie_scores["pre_movie_score"].fillna("SONO NULLO!!") # Sostituiamo
la colonna pre_movie_score nel Df

movie_scores
```

	first_name	last_name	age	sex	pre_movie_score	post_movie_score
0	Tom	Hanks	63.0	m	8.0	10.0
1	NaN	NaN	NaN	NaN	SONO NULLO!!	NaN
2	Hugh	Jackman	51.0	m	SONO NULLO!!	NaN
3	Oprah	Winfrey	66.0	f	6.0	8.0
4	Emma	Stone	31.0	f	7.0	9.0
5	NaN	NaN	NaN	NaN	SONO NULLO!!	NaN
6	NaN	NaN	NaN	NaN	SONO NULLO!!	NaN
7	Emma	Stone	31.0	f	7.0	9.0
8	Kiefer	sutherland	60.0	M	6.0	12.0