

# Progetto di Programmazione Dichiarativa (a.a. 2024/25)

Realizzato da Colli Simone e Merenda Saverio Mattia

Generated by Doxygen 1.9.1



<b>1 Namespace Index</b>	<b>1</b>
1.1 Packages	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Namespace Documentation</b>	<b>7</b>
4.1 main Namespace Reference	7
4.1.1 Function Documentation	7
4.1.1.1 estrazione_dati_per_ssd()	8
4.1.1.2 init()	8
4.1.1.3 init_corsi()	8
4.1.1.4 init_corsi_matricole()	8
4.1.1.5 init_matricole()	8
4.1.1.6 main()	9
4.1.1.7 main_old()	9
4.1.1.8 process_ssd()	9
4.1.1.9 run_tests()	9
4.1.1.10 scrittura_fatti()	9
4.1.1.11 test_estrusione_settori()	10
4.1.1.12 test_filtra_per_colonne()	10
4.1.1.13 test_filtra_per_valori()	10
4.1.1.14 test_parser()	10
4.1.1.15 unisci_file_per_settore()	11
4.1.2 Variable Documentation	11
4.1.2.1 path_coperture	11
4.1.2.2 path_docenti	11
4.1.2.3 path_docenti_a_contratto	11
4.2 modules Namespace Reference	11
4.3 modules.course_parser Namespace Reference	11
4.4 modules.dataset_loader Namespace Reference	12
4.5 modules.dataset_manager Namespace Reference	12
4.6 modules.department_parser Namespace Reference	12
4.7 sanitize Namespace Reference	12
4.7.1 Function Documentation	12
4.7.1.1 aggiorna_cod_tipo_corso()	13
4.7.1.2 compute_extra_data()	13
4.7.1.3 compute_remained()	13
4.7.1.4 sanitize()	13
4.7.1.5 sanitize_codici_corso()	13

4.7.1.6 <code>sanitize_coperture()</code> . . . . .	13
4.7.1.7 <code>sanitize_docenti()</code> . . . . .	13
4.7.2 Variable Documentation . . . . .	13
4.7.2.1 <code>path_coperture</code> . . . . .	14
4.7.2.2 <code>path_coperture_contratti</code> . . . . .	14
4.7.2.3 <code>path_coperture_rimaste</code> . . . . .	14
4.7.2.4 <code>path_docenti</code> . . . . .	14
4.7.2.5 <code>path_ns_coperture</code> . . . . .	14
4.7.2.6 <code>path_ns_docenti</code> . . . . .	14
<b>5 Class Documentation</b> . . . . .	<b>15</b>
5.1 CourseParser Class Reference . . . . .	15
5.1.1 Constructor & Destructor Documentation . . . . .	15
5.1.1.1 <code>__init__()</code> . . . . .	15
5.1.2 Member Function Documentation . . . . .	15
5.1.2.1 <code>add_courses()</code> . . . . .	16
5.1.2.2 <code>parse()</code> . . . . .	16
5.1.3 Member Data Documentation . . . . .	16
5.1.3.1 <code>parser</code> . . . . .	16
5.2 DatasetLoader Class Reference . . . . .	16
5.2.1 Detailed Description . . . . .	17
5.2.2 Constructor & Destructor Documentation . . . . .	17
5.2.2.1 <code>__init__()</code> . . . . .	17
5.2.3 Member Function Documentation . . . . .	17
5.2.3.1 <code>df_to_dict()</code> . . . . .	17
5.2.3.2 <code>filter_by_values()</code> . . . . .	18
5.2.3.3 <code>get_values()</code> . . . . .	18
5.2.3.4 <code>save_to_file()</code> . . . . .	19
5.2.4 Member Data Documentation . . . . .	19
5.2.4.1 <code>input_file_path</code> . . . . .	19
5.3 DatasetManager Class Reference . . . . .	19
5.3.1 Detailed Description . . . . .	19
5.3.2 Constructor & Destructor Documentation . . . . .	20
5.3.2.1 <code>__init__()</code> . . . . .	20
5.3.3 Member Function Documentation . . . . .	20
5.3.3.1 <code>get_courses()</code> . . . . .	20
5.3.3.2 <code>get_departments()</code> . . . . .	20
5.3.3.3 <code>get_professors()</code> . . . . .	20
5.3.3.4 <code>get_sectors()</code> . . . . .	21
5.3.3.5 <code>load_data()</code> . . . . .	21
5.3.3.6 <code>scrivi_coperture()</code> . . . . .	21
5.3.3.7 <code>scrivi_docenti()</code> . . . . .	22

5.3.3.8 <code>scrivi_docenti_a_contratto()</code>	22
5.3.4 Member Data Documentation	22
5.3.4.1 <code>dataset_path</code>	22
5.4 DepartmentParser Class Reference	22
5.4.1 Constructor & Destructor Documentation	23
5.4.1.1 <code>__init__()</code>	23
5.4.2 Member Function Documentation	23
5.4.2.1 <code>add_departments()</code>	23
5.4.2.2 <code>parse()</code>	23
5.4.3 Member Data Documentation	23
5.4.3.1 <code>parser</code>	23
<b>6 File Documentation</b>	<b>25</b>
6.1 <code>src/main.py</code> File Reference	25
6.2 <code>src/modules/__init__.py</code> File Reference	26
6.3 <code>src/modules/course_parser.py</code> File Reference	26
6.4 <code>src/modules/dataset_loader.py</code> File Reference	26
6.5 <code>src/modules/dataset_manager.py</code> File Reference	26
6.6 <code>src/modules/department_parser.py</code> File Reference	26
6.7 <code>src/sanitize.py</code> File Reference	27
<b>Index</b>	<b>29</b>



# Chapter 1

## Namespace Index

### 1.1 Packages

Here are the packages with brief descriptions (if available):

<a href="#">main</a>	7
<a href="#">modules</a>	11
<a href="#">modules.course_parser</a>	11
<a href="#">modules.dataset_loader</a>	12
<a href="#">modules.dataset_manager</a>	12
<a href="#">modules.department_parser</a>	12
<a href="#">sanitize</a>	12





## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">CourseParser</a>	15
<a href="#">DatasetLoader</a>	16
<a href="#">DatasetManager</a>	19
<a href="#">DepartmentParser</a>	22



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

src/main.py . . . . .	25
src/sanitize.py . . . . .	27
src/modules/__init__.py . . . . .	26
src/modules/course_parser.py . . . . .	26
src/modules/dataset_loader.py . . . . .	26
src/modules/dataset_manager.py . . . . .	26
src/modules/department_parser.py . . . . .	26



## Chapter 4

# Namespace Documentation

### 4.1 main Namespace Reference

#### Functions

- def [estrazione\\_dati\\_per\\_ssd](#) (path=[path\\_coperture](#))
  - def [init](#) ()
  - def [init\\_corsi](#) (filename)
  - def [init\\_corsi\\_matricole](#) (filepathCorsi, filepathProf)
  - def [init\\_matricole](#) (filename)
  - def [main](#) ()
  - def [main\\_old](#) ()
  - def [process\\_ssd](#) (dataset\_loader, ssd, failed\_ssds)
  - def [run\\_tests](#) ()
  - def [scrittura\\_fatti](#) (filters=None, file\_name=None, path=[path\\_coperture](#))
- FUNZIONI PRINCIPALI #####.*
- def [test\\_estrazione\\_settori](#) ()
  - def [test\\_filtra\\_per\\_colonne](#) ()
  - def [test\\_filtra\\_per\\_valori](#) ()
  - def [test\\_parser](#) ()
- TEST #####.*
- def [unisci\\_file\\_per\\_settore](#) (path)

#### Variables

- string [path\\_coperture](#) = "dataset/coperture.xlsx"
- VARIABILI GLOBALI #####.*
- string [path\\_docenti](#) = "dataset/docenti.xlsx"
  - string [path\\_docenti\\_a\\_contratto](#) = "dataset/docenti\_a\_contratto.xlsx"

#### 4.1.1 Function Documentation

#### 4.1.1.1 estrazione\_dati\_per\_ssd()

```
def main.estrusione_dati_per_ssd (
    path = path_coperture )
```

Estrae i dati per ogni SSD, applicando il filtro e salvando i file CSV nella cartella 'dataset/ssd/'.

Questa funzione esegue le seguenti operazioni:

1. Carica i dati da un file Excel e seleziona la colonna 'SSD', rimuovendo i duplicati.
2. Crea una barra di progresso per monitorare l'elaborazione degli SSD.
3. Utilizza un 'ThreadPoolExecutor' per elaborare i dati in parallelo, migliorando le performance.
4. Crea la cartella di output 'dataset/ssd/' se non esiste già.
5. Per ogni SSD, chiama la funzione 'process\_ssd' per applicare il filtro e salvare il file CSV corrispondente.
6. Gestisce gli errori durante l'elaborazione e registra gli SSD che non sono stati salvati.

:param dataset\_loader: Istanza di 'DatasetLoader' per accedere ai metodi di filtraggio e salvataggio.  
:param failed\_ssds: Set per raccogliere gli SSD che causano errori durante l'elaborazione.

#### 4.1.1.2 init()

```
def main.init ( )
```

#### 4.1.1.3 init\_corsi()

```
def main.init_corsi (
    filename )
```

#### 4.1.1.4 init\_corsi\_matricole()

```
def main.init_corsi_matricole (
    filepathCorsi,
    filepathProf )
```

#### 4.1.1.5 init\_matricole()

```
def main.init_matricole (
    filename )
```

#### 4.1.1.6 main()

```
def main.main ( )
```

#### 4.1.1.7 main\_old()

```
def main.main_old ( )
```

#### 4.1.1.8 process\_ssd()

```
def main.process_ssd (
    dataset_loader,
    ssd,
    failed_ssds )
```

Elabora un singolo valore SSD, applicando il filtro e salvando il file.  
:param dataset\_loader: Istanza di DatasetLoader.  
:param ssd: Valore dell'SSD da elaborare.  
:param failed\_ssds: Set per raccogliere gli SSD che causano errori.

#### 4.1.1.9 run\_tests()

```
def main.run_tests ( )
```

#### 4.1.1.10 scrittura\_fatti()

```
def main.scrittura_fatti (
    filters = None,
    file_name = None,
    path = path\_coperture )
```

FUNZIONI PRINCIPALI #####.

Filtra i dati da un dataset Excel basandosi sui criteri forniti e scrive i fatti in un file ASP.

```
:param filters: Dizionario contenente i criteri di filtraggio.
Esempio:
{
    'Cod. Tipo Corso': ['LM'],
    'SSD': ['INF/01']
}
Se non specificato, vengono utilizzati i valori predefiniti:
{'Cod. Tipo Corso': ['LM'], 'SSD': ['INF/01']}.
:param file_name: Nome del file di output (senza estensione) dove verranno salvati i fatti generati.
Se non specificato, il file verrà salvato con il nome predefinito "test".
```

Funzionamento:

1. Inizializza un'istanza di 'DatasetLoader' per caricare i dati dal file 'dataset/coperture.xlsx'.
2. Filtra i dati in base ai criteri specificati in 'filters'.
3. Inizializza un'istanza di 'DatasetManager'.
4. Utilizza il metodo 'write\_atoms' del 'DatasetManager' per scrivere i fatti filtrati nel file ASP specificato.

Esempio:

```
-----
Se il dataset contiene corsi con SSD 'INF/01' e tipo corso 'LM', e il filtro fornito è:
filters = {
    'Cod. Tipo Corso': ['LM'],
    'SSD': ['INF/01']
}
e 'file_name = "informatica"', i dati filtrati verranno salvati nel file 'informatica.lp'.
```

```
:raises FileNotFoundError: Se il file specificato nel DatasetLoader non esiste.
:raises Exception: Per altri errori durante il filtraggio o la scrittura dei fatti.
```

#### 4.1.1.11 test\_estrazione\_settori()

```
def main.test_estrazione_settori ( )
```

#### 4.1.1.12 test\_filtra\_per\_colonne()

```
def main.test_filtra_per_colonne ( )
```

#### 4.1.1.13 test\_filtra\_per\_valori()

```
def main.test_filtra_per_valori ( )
```

#### 4.1.1.14 test\_parser()

```
def main.test_parser ( )
```

TEST #####.



#### 4.1.1.15 unisci\_file\_per\_settore()

```
def main.unisci_file_per_settore (
    path )
```

Unisce i file CSV che hanno lo stesso settore in base al prefisso del nome file.  
Salva il risultato in un unico file per settore senza duplicare l'intestazione.  
Sposta i file originali con '\_' nel nome nella cartella 'caratterizzanti'.

:param path: Percorso alla cartella contenente i file CSV.

### 4.1.2 Variable Documentation

#### 4.1.2.1 path\_coperture

```
string path_coperture = "dataset/coperture.xlsx"
```

VARIABILI GLOBALI #####.

#### 4.1.2.2 path\_docenti

```
string path_docenti = "dataset/docenti.xlsx"
```

#### 4.1.2.3 path\_docenti\_a\_contratto

```
string path_docenti_a_contratto = "dataset/docenti_a_contratto.xlsx"
```

## 4.2 modules Namespace Reference

### Namespaces

- [course\\_parser](#)
- [dataset\\_loader](#)
- [dataset\\_manager](#)
- [department\\_parser](#)

## 4.3 modules.course\_parser Namespace Reference

### Classes

- class [CourseParser](#)

## 4.4 modules.dataset\_loader Namespace Reference

### Classes

- class [DatasetLoader](#)

## 4.5 modules.dataset\_manager Namespace Reference

### Classes

- class [DatasetManager](#)

## 4.6 modules.department\_parser Namespace Reference

### Classes

- class [DepartmentParser](#)

## 4.7 sanitize Namespace Reference

### Functions

- def [aggiorna\\_cod\\_tipo\\_corso](#) (row)
- def [compute\\_extra\\_data](#) ()
- def [compute\\_remained](#) ()
- def [sanitize](#) ()
- def [sanitize\\_codici\\_corso](#) (df)
- def [sanitize\\_coperture](#) ()
- def [sanitize\\_docenti](#) ()

### Variables

- string [path\\_coperture](#) = "dataset/coperture.xlsx"
- string [path\\_coperture\\_contratti](#) = "dataset/docenti\_a\_contratto.xlsx"
- string [path\\_coperture\\_rimaste](#) = "dataset/insegnamenti\_senza\_docente.xlsx"
- string [path\\_docenti](#) = "dataset/docenti.xlsx"
- string [path\\_ns\\_coperture](#) = "dataset/originali/coperture.xlsx"
- string [path\\_ns\\_docenti](#) = "dataset/originali/docenti.xlsx"

### 4.7.1 Function Documentation

#### 4.7.1.1 aggiorna\_cod\_tipo\_corso()

```
def sanitize.aggiorna_cod_tipo_corso (
    row )
```

#### 4.7.1.2 compute\_extra\_data()

```
def sanitize.compute_extra_data ( )
```

#### 4.7.1.3 compute\_remained()

```
def sanitize.compute_remained ( )
```

#### 4.7.1.4 sanitize()

```
def sanitize.sanitize ( )
```

#### 4.7.1.5 sanitize\_codici\_corso()

```
def sanitize.sanitize_codici_corso (
    df )
```

#### 4.7.1.6 sanitize\_coperture()

```
def sanitize.sanitize_coperture ( )
```

#### 4.7.1.7 sanitize\_docenti()

```
def sanitize.sanitize_docenti ( )
```

### 4.7.2 Variable Documentation

#### 4.7.2.1 path\_coperture

```
string path_coperture = "dataset/coperture.xlsx"
```

#### 4.7.2.2 path\_coperture\_contratti

```
string path_coperture_contratti = "dataset/docenti_a_contratto.xlsx"
```

#### 4.7.2.3 path\_coperture\_rimaste

```
string path_coperture_rimaste = "dataset/insegnamenti_senza_docente.xlsx"
```

#### 4.7.2.4 path\_docenti

```
string path_docenti = "dataset/docenti.xlsx"
```

#### 4.7.2.5 path\_ns\_coperture

```
string path_ns_coperture = "dataset/originali/coperture.xlsx"
```

#### 4.7.2.6 path\_ns\_docenti

```
string path_ns_docenti = "dataset/originali/docenti.xlsx"
```

## Chapter 5

# Class Documentation

### 5.1 CourseParser Class Reference

#### Public Member Functions

- def [\\_\\_init\\_\\_](#) (self)
- def [add\\_courses](#) (self, courses)
- def [parse](#) (self)

#### Public Attributes

- [parser](#)

#### 5.1.1 Constructor & Destructor Documentation

##### 5.1.1.1 `__init__()`

```
def __init__ (  
    self )
```

Inizializza il parser per i corsi.

#### 5.1.2 Member Function Documentation

#### 5.1.2.1 add\_courses()

```
def add_courses (
    self,
    courses )
```

Aggiunge argomenti dinamici per ciascun dipartimento.  
:param departments: Lista dei corsi.

#### 5.1.2.2 parse()

```
def parse (
    self )
```

Effettua il parsing degli argomenti.  
:return: Gli argomenti parsati.

### 5.1.3 Member Data Documentation

#### 5.1.3.1 parser

```
parser
```

The documentation for this class was generated from the following file:

- src/modules/[course\\_parser.py](#)

## 5.2 DatasetLoader Class Reference

### Public Member Functions

- def [\\_\\_init\\_\\_](#) (self, [input\\_file\\_path](#)="dataset/coperture.xlsx")
- def [df\\_to\\_dict](#) (self, df, id\_type="col")
- def [filter\\_by\\_values](#) (self, filters, dataset=None, only\_prefix=False)
- def [get\\_values](#) (self, dataset=None, columns=None)
- def [save\\_to\\_file](#) (self, df, output\_file\_path, file\_format="csv")

### Public Attributes

- [input\\_file\\_path](#)

## 5.2.1 Detailed Description

Classe per la gestione del caricamento, filtraggio e salvataggio dei dati da file Excel o CSV.

## 5.2.2 Constructor & Destructor Documentation

### 5.2.2.1 \_\_init\_\_()

```
def __init__ (
    self,
    input_file_path = "dataset/coperture.xlsx" )
```

Inizializza la classe DatasetLoader con il percorso al dataset.

:param input\_file\_path: Path relativo al file Excel da leggere (default: "dataset/coperture.xlsx").

## 5.2.3 Member Function Documentation

### 5.2.3.1 df\_to\_dict()

```
def df_to_dict (
    self,
    df,
    id_type = "col" )
```

Converte un DataFrame Pandas in un dizionario strutturato.

Il dizionario risultante contiene:

- "rows": lista degli identificativi delle righe (index del DataFrame).
- "cols": lista degli identificativi delle colonne (header del DataFrame).
- Chiavi dinamiche: il contenuto delle colonne (default) o delle righe.

:param df: Un DataFrame Pandas da convertire.

:type df: pandas.DataFrame

:param id\_type: Tipo di identificatore per il contenuto. "col" per colonne (default),  
"row" per righe.

:type id\_type: str, optional

:return: Un dizionario strutturato contenente informazioni su righe, colonne  
e i dati associati.

:rtype: dict

:example:

input:

```
df =
  A  B  C
0  1  2  3
1  4  5  6
```

output (id\_type="col"):

```
{
    "rows": [0, 1],
    "cols": ["A", "B", "C"],
```

```
        "A": [1, 4],
        "B": [2, 5],
        "C": [3, 6]
    }

    output (id_type="row"):
    {
        "rows": [0, 1],
        "cols": ["A", "B", "C"],
        0: [1, 2, 3],
        1: [4, 5, 6]
    }
```

### 5.2.3.2 filter\_by\_values()

```
def filter_by_values (
    self,
    filters,
    dataset = None,
    only_prefix = False )
```

Filtra i dati in base a più valori specifici in diverse colonne, convertendo tutti i valori in stringhe.

:param filters: Dizionario in cui le chiavi sono i nomi delle colonne e i valori sono liste di valori da filtrare.  
Esempio: {'Cod. Tipo Corso': ['LM'], 'SSD': ['INF/01', 'MAT/05']}.

:param dataset: DataFrame esistente da utilizzare (default: None). Se None, viene caricato il file Excel.

:return: DataFrame contenente solo le righe che soddisfano i criteri di filtraggio.

:raises ImportError: Se il pacchetto openpyxl non è installato.

:raises KeyError: Se una delle colonne specificate nei filtri non esiste nel dataset.

:raises Exception: Per altri errori durante il filtraggio.

### 5.2.3.3 get\_values()

```
def get_values (
    self,
    dataset = None,
    columns = None )
```

Carica un file Excel o utilizza un dataset esistente, filtrando opzionalmente per colonne specifiche.

:param dataset: DataFrame esistente da utilizzare (default: None). Se None, viene caricato il file Excel.

:param columns: Lista di colonne da mantenere (esempio: ['Matricola', 'Cognome']).

:return: DataFrame contenente tutte le righe, filtrato per le colonne specificate (se fornite).

:raises ImportError: Se il pacchetto openpyxl non è installato.

:raises KeyError: Se una o più colonne specificate non esistono nel dataset.



#### 5.2.3.4 save\_to\_file()

```
def save_to_file (
    self,
    df,
    output_file_path,
    file_format = "csv" )
```

Salva il DataFrame su un file specificato.

:param df: Il DataFrame da salvare.

:param output\_file\_path: Percorso del file di output.

:param file\_format: Formato del file, può essere 'csv' o 'excel'.

:raises ValueError: Se il formato specificato non è 'csv' o 'excel'.

:raises Exception: Per altri errori durante il salvataggio del file.

### 5.2.4 Member Data Documentation

#### 5.2.4.1 input\_file\_path

input\_file\_path

The documentation for this class was generated from the following file:

- [src/modules/dataset\\_loader.py](#)

## 5.3 DatasetManager Class Reference

### Public Member Functions

- def [\\_\\_init\\_\\_](#) (self, [dataset\\_path](#)="dataset/")
- def [get\\_courses](#) (self)
- def [get\\_departments](#) (self)
- def [get\\_professors](#) (self)
- def [get\\_sectors](#) (self, SSD=None)
- def [load\\_data](#) (self, filename)
- def [scrivi\\_copertura](#) (self, df, filename)
- def [scrivi\\_docenti](#) (self, df, filename)
- def [scrivi\\_docenti\\_a\\_contratto](#) (self, df, filename)

### Public Attributes

- [dataset\\_path](#)

#### 5.3.1 Detailed Description

Classe per la gestione di file di dataset e la generazione di file ASP strutturati.

## 5.3.2 Constructor & Destructor Documentation

### 5.3.2.1 `__init__()`

```
def __init__ (
    self,
    dataset_path = "dataset/" )
```

Inizializza la classe DatasetManager con il percorso della cartella contenente i file di dataset.  
:param dataset\_path: Path alla cartella contenente i file di dataset (default: "dataset/").

## 5.3.3 Member Function Documentation

### 5.3.3.1 `get_courses()`

```
def get_courses (
    self )
```

Ottiene un dizionario {"codice\_corso": "corso"} basandosi sui file presenti nella cartella del dataset.  
:return: Un dizionario di corsi.  
:raises FileNotFoundError: Se la cartella del dataset non esiste.

### 5.3.3.2 `get_departments()`

```
def get_departments (
    self )
```

Ottiene la lista dei dipartimenti basandosi sui file presenti nella cartella del dataset.  
:return: Una lista di dipartimenti (nomi dei file senza estensione).  
:raises FileNotFoundError: Se la cartella del dataset non esiste.

### 5.3.3.3 `get_professors()`

```
def get_professors (
    self )
```

Ottiene un dizionario {"codice\_corso": {"matricola"}} basandosi sui file presenti nella cartella del dataset.  
:return: Un dizionario di corsi: set\_matricole.  
:raises FileNotFoundError: Se la cartella del dataset non esiste.

#### 5.3.3.4 get\_sectors()

```
def get_sectors (
    self,
    SSD = None )
```

Ottiene la lista dei settori partendo da una lista di SSD.

Ogni SSD è rappresentato nel formato "SETTORE/CODICE". Questa funzione estrae e restituisce solo la parte relativa al settore (prima del separatore '/').

```
:param SSD: Una lista di stringhe rappresentanti i SSD. Ad esempio, ["INF/01", "MAT/03"].
:type SSD: list[str], optional
:return: Un set di settori se la lista passata come parametro è non vuota.
        Ritorna un set vuoto se il parametro è None o vuoto.
:rtype: set[str]

:example:
input: ["INF/01", "MAT/03", "FIS/07"]
output: {"INF", "MAT", "FIS"}
```

#### 5.3.3.5 load\_data()

```
def load_data (
    self,
    filename )
```

Legge i dati da un file di testo e li stampa.

```
:param filename: Nome del file (senza estensione) da cui leggere i dati.
:raises FileNotFoundError: Se il file specificato non esiste.
:raises Exception: Per altri errori durante la lettura del file.
```

#### 5.3.3.6 scrivi\_coperture()

```
def scrivi_coperture (
    self,
    df,
    filename )
```

Scrivi i dati del DataFrame in un file ASP (.lp), organizzandoli per sezioni con eliminazione di duplicati.

```
:param df: DataFrame contenente i dati da salvare.
:param filename: Nome del file di output (senza estensione).
:raises Exception: Per errori durante la scrittura del file.
```

### 5.3.3.7 `scrivi_docenti()`

```
def scrivi_docenti (
    self,
    df,
    filename )
```

Scrivi i dati del DataFrame in un file ASP (.lp), organizzandoli per sezioni con eliminazione di duplicati.  
:param df: DataFrame contenente i dati da salvare.  
:param filename: Nome del file di output (senza estensione).  
:raises Exception: Per errori durante la scrittura del file.

### 5.3.3.8 `scrivi_docenti_a_contratto()`

```
def scrivi_docenti_a_contratto (
    self,
    df,
    filename )
```

Scrivi i dati del DataFrame in un file ASP (.lp), organizzandoli per sezioni con eliminazione di duplicati.  
:param df: DataFrame contenente i dati da salvare.  
:param filename: Nome del file di output (senza estensione).  
:raises Exception: Per errori durante la scrittura del file.

## 5.3.4 Member Data Documentation

### 5.3.4.1 `dataset_path`

`dataset_path`

The documentation for this class was generated from the following file:

- [src/modules/dataset\\_manager.py](#)

## 5.4 DepartmentParser Class Reference

### Public Member Functions

- `def \_\_init\_\_ (self)`
- `def add\_departments (self, departments)`
- `def parse (self)`

## Public Attributes

- [parser](#)

## 5.4.1 Constructor & Destructor Documentation

### 5.4.1.1 `__init__()`

```
def __init__ (
    self )
```

Inizializza il parser per i dipartimenti.

## 5.4.2 Member Function Documentation

### 5.4.2.1 `add_departments()`

```
def add_departments (
    self,
    departments )
```

Aggiunge argomenti dinamici per ciascun dipartimento.  
:param departments: Lista dei dipartimenti.

### 5.4.2.2 `parse()`

```
def parse (
    self )
```

Effettua il parsing degli argomenti.  
:return: Gli argomenti parsati.

## 5.4.3 Member Data Documentation

### 5.4.3.1 `parser`

`parser`

The documentation for this class was generated from the following file:

- `src/modules/department\_parser.py`



## Chapter 6

# File Documentation

### 6.1 src/main.py File Reference

#### Namespaces

- [main](#)

#### Functions

- def [estrazione\\_dati\\_per\\_ssd](#) (path=path\_coperture)
- def [init](#) ()
- def [init\\_corsi](#) (filename)
- def [init\\_corsi\\_matricole](#) (filepathCorsi, filepathProf)
- def [init\\_matricole](#) (filename)
- def [main](#) ()
- def [main\\_old](#) ()
- def [process\\_ssd](#) (dataset\_loader, ssd, failed\_ssds)
- def [run\\_tests](#) ()
- def [scrittura\\_fatti](#) (filters=None, file\_name=None, path=path\_coperture)  
*FUNZIONI PRINCIPALI #####.*
- def [test\\_estrazione\\_settori](#) ()
- def [test\\_filtra\\_per\\_colonne](#) ()
- def [test\\_filtra\\_per\\_valori](#) ()
- def [test\\_parser](#) ()  
*TEST #####.*
- def [unisci\\_file\\_per\\_settore](#) (path)

#### Variables

- string [path\\_coperture](#) = "dataset/coperture.xlsx"  
*VARIABILI GLOBALI #####.*
- string [path\\_docenti](#) = "dataset/docenti.xlsx"
- string [path\\_docenti\\_a\\_contratto](#) = "dataset/docenti\_a\_contratto.xlsx"

## 6.2 src/modules/\_\_init\_\_.py File Reference

### Namespaces

- [modules](#)

## 6.3 src/modules/course\_parser.py File Reference

### Classes

- class [CourseParser](#)

### Namespaces

- [modules.course\\_parser](#)

## 6.4 src/modules/dataset\_loader.py File Reference

### Classes

- class [DatasetLoader](#)

### Namespaces

- [modules.dataset\\_loader](#)

## 6.5 src/modules/dataset\_manager.py File Reference

### Classes

- class [DatasetManager](#)

### Namespaces

- [modules.dataset\\_manager](#)

## 6.6 src/modules/department\_parser.py File Reference

### Classes

- class [DepartmentParser](#)



## Namespaces

- [modules.department\\_parser](#)

## 6.7 src/sanitize.py File Reference

### Namespaces

- [sanitize](#)

### Functions

- def [aggiorna\\_cod\\_tipo\\_corso](#) (row)
- def [compute\\_extra\\_data](#) ()
- def [compute\\_remained](#) ()
- def [sanitize](#) ()
- def [sanitize\\_codici\\_corso](#) (df)
- def [sanitize\\_coperture](#) ()
- def [sanitize\\_docenti](#) ()

### Variables

- string [path\\_coperture](#) = "dataset/coperture.xlsx"
- string [path\\_coperture\\_contratti](#) = "dataset/docenti\_a\_contratto.xlsx"
- string [path\\_coperture\\_rimaste](#) = "dataset/insegnamenti\_senza\_docente.xlsx"
- string [path\\_docenti](#) = "dataset/docenti.xlsx"
- string [path\\_ns\\_coperture](#) = "dataset/originali/coperture.xlsx"
- string [path\\_ns\\_docenti](#) = "dataset/originali/docenti.xlsx"



# Index

- `__init__`
    - `CourseParser`, 15
    - `DatasetLoader`, 17
    - `DatasetManager`, 20
    - `DepartmentParser`, 23
- `add_courses`
  - `CourseParser`, 15
- `add_departments`
  - `DepartmentParser`, 23
- `aggiorna_cod_tipo_corso`
  - `sanitize`, 12
- `compute_extra_data`
  - `sanitize`, 13
- `compute_remained`
  - `sanitize`, 13
- `CourseParser`, 15
  - `__init__`, 15
  - `add_courses`, 15
  - `parse`, 16
  - `parser`, 16
- `dataset_path`
  - `DatasetManager`, 22
- `DatasetLoader`, 16
  - `__init__`, 17
  - `df_to_dict`, 17
  - `filter_by_values`, 18
  - `get_values`, 18
  - `input_file_path`, 19
  - `save_to_file`, 18
- `DatasetManager`, 19
  - `__init__`, 20
  - `dataset_path`, 22
  - `get_courses`, 20
  - `get_departments`, 20
  - `get_professors`, 20
  - `get_sectors`, 20
  - `load_data`, 21
  - `scrivi_copertura`, 21
  - `scrivi_docenti`, 21
  - `scrivi_docenti_a_contratto`, 22
- `DepartmentParser`, 22
  - `__init__`, 23
  - `add_departments`, 23
  - `parse`, 23
  - `parser`, 23
- `df_to_dict`
  - `DatasetLoader`, 17
- `estrazione_dati_per_ssd`
  - `main`, 7
- `filter_by_values`
  - `DatasetLoader`, 18
- `get_courses`
  - `DatasetManager`, 20
- `get_departments`
  - `DatasetManager`, 20
- `get_professors`
  - `DatasetManager`, 20
- `get_sectors`
  - `DatasetManager`, 20
- `get_values`
  - `DatasetLoader`, 18
- `init`
  - `main`, 8
- `init_corsi`
  - `main`, 8
- `init_corsi_matricole`
  - `main`, 8
- `init_matricole`
  - `main`, 8
- `input_file_path`
  - `DatasetLoader`, 19
- `load_data`
  - `DatasetManager`, 21
- `main`, 7
  - `estrazione_dati_per_ssd`, 7
  - `init`, 8
  - `init_corsi`, 8
  - `init_corsi_matricole`, 8
  - `init_matricole`, 8
  - `main`, 8
  - `main_old`, 9
  - `path_copertura`, 11
  - `path_docenti`, 11
  - `path_docenti_a_contratto`, 11
  - `process_ssd`, 9
  - `run_tests`, 9
  - `scrittura_fatti`, 9
  - `test_estrazione_settori`, 10
  - `test_filtra_per_colonne`, 10
  - `test_filtra_per_valori`, 10
  - `test_parser`, 10
  - `unisci_file_per_settore`, 10
- `main_old`

- main, [9](#)
- modules, [11](#)
- modules.course\_parser, [11](#)
- modules.dataset\_loader, [12](#)
- modules.dataset\_manager, [12](#)
- modules.department\_parser, [12](#)
- parse
  - CourseParser, [16](#)
  - DepartmentParser, [23](#)
- parser
  - CourseParser, [16](#)
  - DepartmentParser, [23](#)
- path\_coperture
  - main, [11](#)
  - sanitize, [13](#)
- path\_coperture\_contratti
  - sanitize, [14](#)
- path\_coperture\_rimaste
  - sanitize, [14](#)
- path\_docenti
  - main, [11](#)
  - sanitize, [14](#)
- path\_docenti\_a\_contratto
  - main, [11](#)
- path\_ns\_coperture
  - sanitize, [14](#)
- path\_ns\_docenti
  - sanitize, [14](#)
- process\_ssd
  - main, [9](#)
- run\_tests
  - main, [9](#)
- sanitize, [12](#)
  - aggiorna\_cod\_tipo\_corso, [12](#)
  - compute\_extra\_data, [13](#)
  - compute\_remained, [13](#)
  - path\_coperture, [13](#)
  - path\_coperture\_contratti, [14](#)
  - path\_coperture\_rimaste, [14](#)
  - path\_docenti, [14](#)
  - path\_ns\_coperture, [14](#)
  - path\_ns\_docenti, [14](#)
  - sanitize, [13](#)
  - sanitize\_codici\_corso, [13](#)
  - sanitize\_coperture, [13](#)
  - sanitize\_docenti, [13](#)
- sanitize\_codici\_corso
  - sanitize, [13](#)
- sanitize\_coperture
  - sanitize, [13](#)
- sanitize\_docenti
  - sanitize, [13](#)
- save\_to\_file
  - DatasetLoader, [18](#)
- scrittura\_fatti
  - main, [9](#)
- scrivi\_coperture
  - DatasetManager, [21](#)
- scrivi\_docenti
  - DatasetManager, [21](#)
- scrivi\_docenti\_a\_contratto
  - DatasetManager, [22](#)
- src/main.py, [25](#)
- src/modules/\_\_init\_\_.py, [26](#)
- src/modules/course\_parser.py, [26](#)
- src/modules/dataset\_loader.py, [26](#)
- src/modules/dataset\_manager.py, [26](#)
- src/modules/department\_parser.py, [26](#)
- src/sanitize.py, [27](#)
- test\_estrazione\_settori
  - main, [10](#)
- test\_filtra\_per\_colonne
  - main, [10](#)
- test\_filtra\_per\_valori
  - main, [10](#)
- test\_parser
  - main, [10](#)
- unisci\_file\_per\_settore
  - main, [10](#)