## Document Management

Create the following project structure



### Class Topic

The **Topic** class should receive the following **parameters** upon initialization: **id: int**, **topic: str**, **storage\_folder: str**. It should have **two methods**:

* **edit(new\_topic: str, new\_storage\_folder: str)** – change the **topic** and the **storage folder**
* **\_\_repr\_\_()** – returns a **string representation** of the topic in the format: **"Topic {id}: {topic} in {storage\_folder}"**

### Class Category

The **Category** class should receive the following **parameters** upon initialization: **id: int**, **name: str**. The class should have **two methods**:

* **edit(new\_name: str)** – edit the **name** of the category
* **\_\_repr\_\_()** – returns a **string representation** of the category in the following format: **"Category {id}: {name}"**

### Class Document

The **Document** class should receive the following **parameters** upon initialization: **id: int**, **category\_id: int**, **topic\_id: int**, **file\_name: str**. The class should also have **one more attribute** called **tags** (**empty list** upon initialization). The class should also have **4 methods**:

* **from\_instances(id:int, category:Category, topic:Topic, file\_name:str)** – create a **new instance** using the provided **category** and **topic** instances
* **add\_tag(tag\_content: str)** – if the **tag** is **not** already in the tags **list**, **add** it to the tags list
* **remove\_tag(tag\_content:str)** – if the tag is **in** the tags **list**, **delete it**
* **edit(file\_name:str)** – **change** the **file** **name** with the given one
* **\_\_repr\_\_()** – returns a string representation of a document in the format: **"Document {id}: {file\_name}; category {category\_id}, topic {topic\_id}, tags: {tags joined by comma and space)}"**

### Class Storage

Upon initialization the class **Storage** will **not receive any parameters**. It should have **3 instance attributes**: **categories** (empty list), **topics** (empty list), **documents** (empty list). The class should have the following **methods**:

* **add\_category(category:Category)** – add the category if it **is not in the list**
* **add\_topic(topic:Topic)** – add the topic if it **does not exist**
* **add\_document(document:Document)** – add the document if it **does not exist**
* **edit\_category(category\_id: int, new\_name:str)** – edit the **name** of the category with the provided **id**
* **edit\_topic(topic\_id: int, new\_topic: str, new\_storage\_folder: str)** – edit the **topic** with the given **id**
* **edit\_document(document\_id: int, new\_file\_name: str)** – edit the **document** with the given **id**
* **delete\_category(category\_id)** – delete the **category** with the provided **id**
* **delete\_topic(topic\_id)** – delete the **topic** with the provided **id**
* **delete\_document(document\_id)** – delete the **document** with the provided **id**
* **get\_document(document\_id)** – return the **document** with the provided **id**
* **\_\_repr\_\_()** – returns a **string representation** of each document on **separate lines**

### Examples

|  |
| --- |
| **Test Code** |
| from project.category import Category  from project.document import Document  from project.storage import Storage  from project.topic import Topic  c1 = Category(1, "work")  t1 = Topic(1, "daily tasks", "C:\\work\_documents")  d1 = Document(1, 1, 1, "finilize project")  d1.add\_tag("urgent")  d1.add\_tag("work")  storage = Storage()  storage.add\_category(c1)  storage.add\_topic(t1)  storage.add\_document(d1)  print(c1)  print(t1)  print(storage.get\_document(1))  print(storage) |
| **Output** |
| Category 1: work  Topic 1: daily tasks in C:\work\_documents  Document 1: finilize project; category 1, topic 1, tags: urgent, work  Document 1: finilize project; category 1, topic 1, tags: urgent, work |