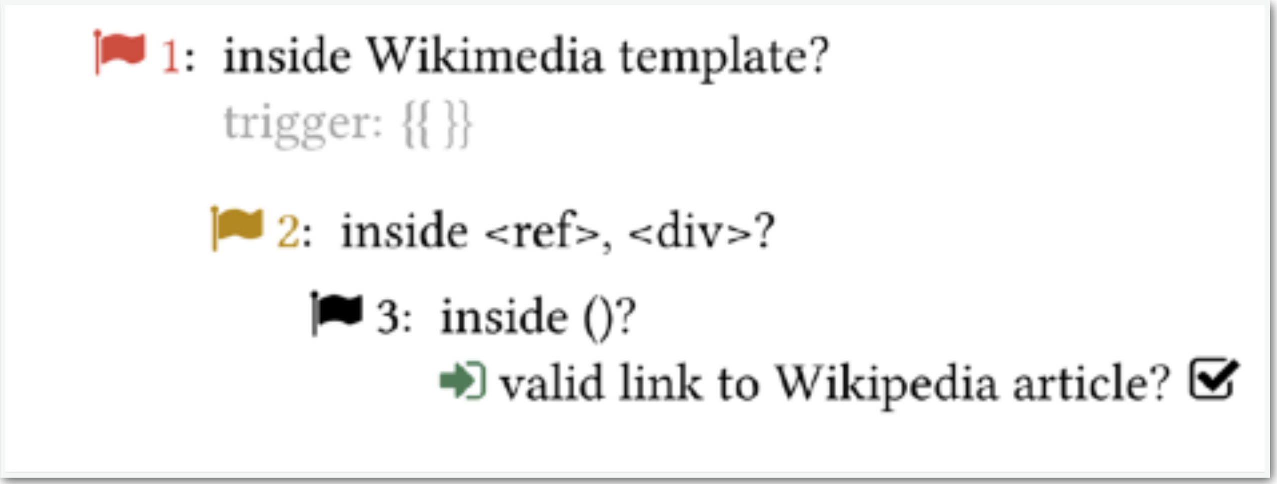
```
df = sqlContext.read.format(‘com.databricks.spark.xml')
      .options(rowTag='page').load(full_xml)
```

```
network_df = df.select(“title”, “revision").map(lambda s:
     (s[0],Article(s[1].text[0]).first_link())).collect()
```



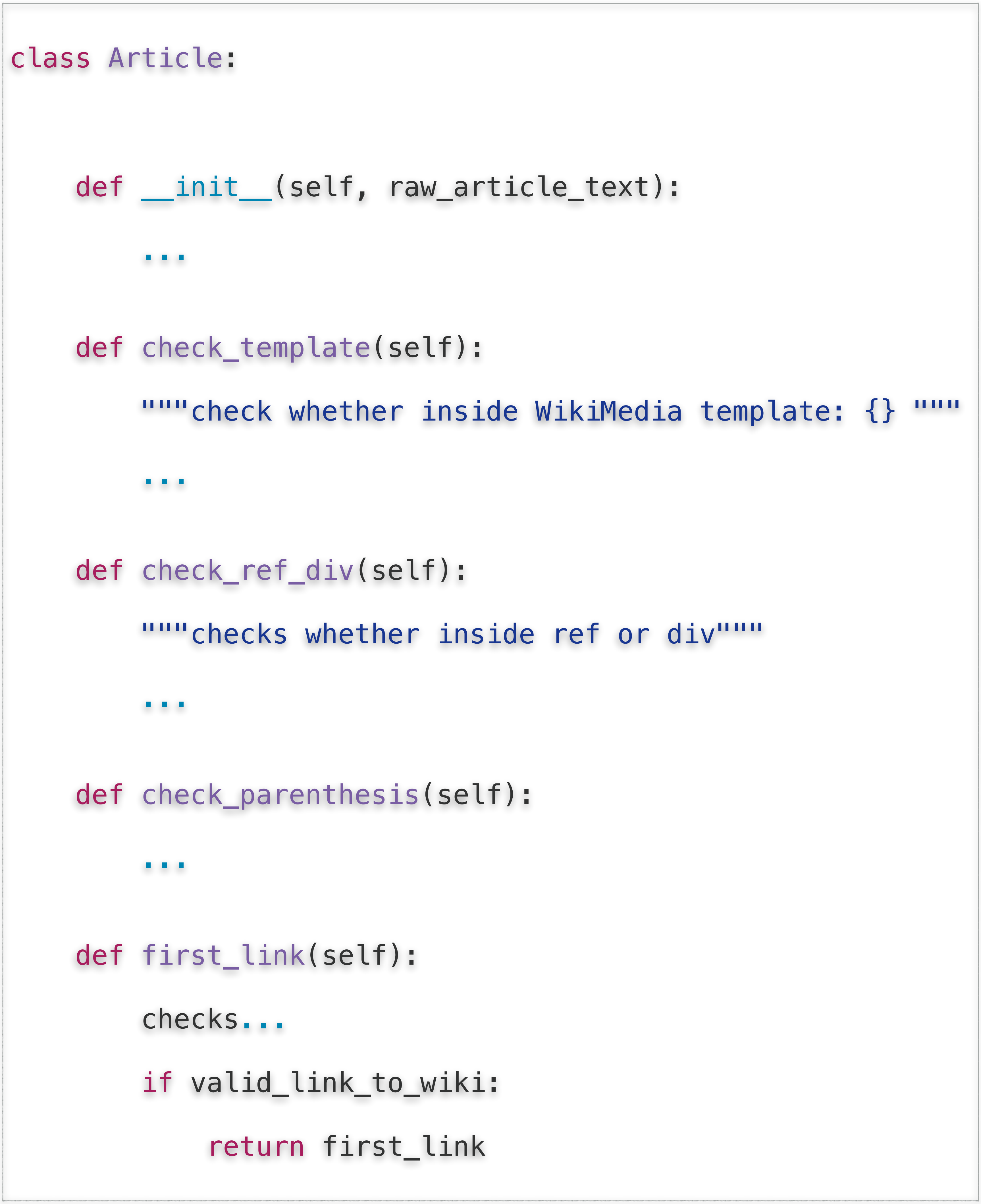
1: inside Wikimedia template?

trigger: {{ }}

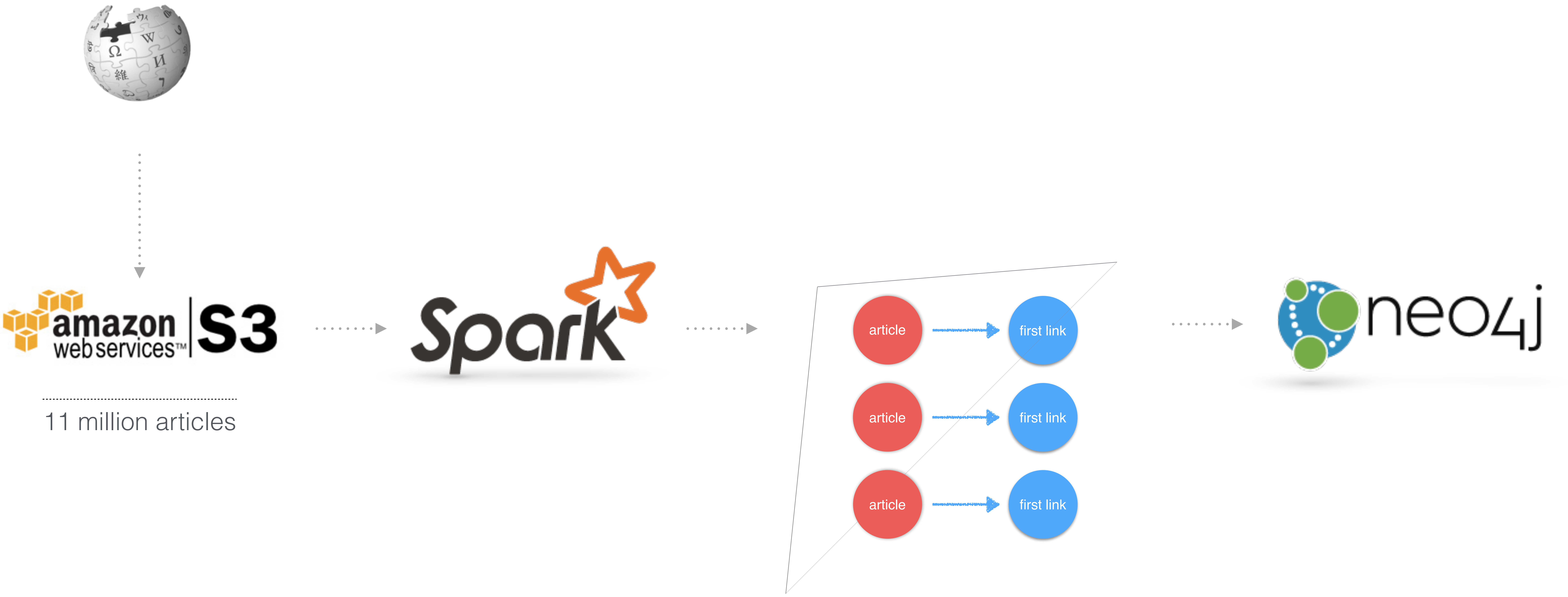
2: inside <ref>, <div>?

3: inside ()?

valid link to Wikipedia article? ☑



```
class Article:

    def __init__(self, raw_article_text):
        ...

    def check_template(self):
        """check whether inside WikiMedia template: {} """
        ...

    def check_ref_div(self):
        """checks whether inside ref or div"""
        ...

    def check_parenthesis(self):
        ...

    def first_link(self):
        checks...
        if valid_link_to_wiki:
            return first_link
```

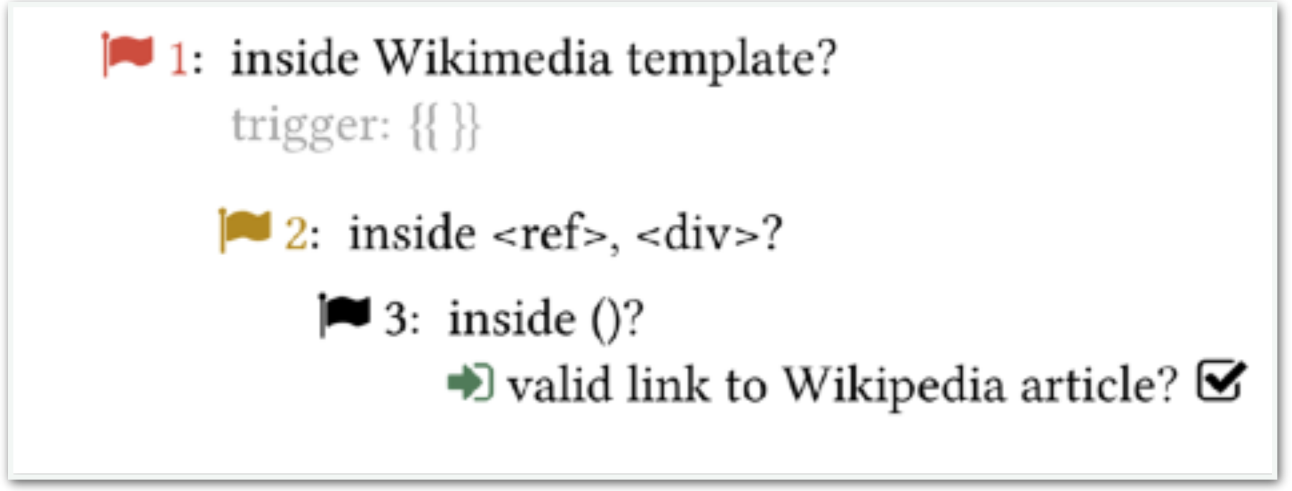


11 million articles

article
first link
article
first link
article
first link

```
df = sqlContext.read.format(‘com.databricks.spark.xml')
       .options(rowTag='page').load(full_xml)
```

```
network_df = df.select(“title”, “revision").map(lambda s:
     (s[0],Article(s[1].text[0]).first_link())).collect()
```

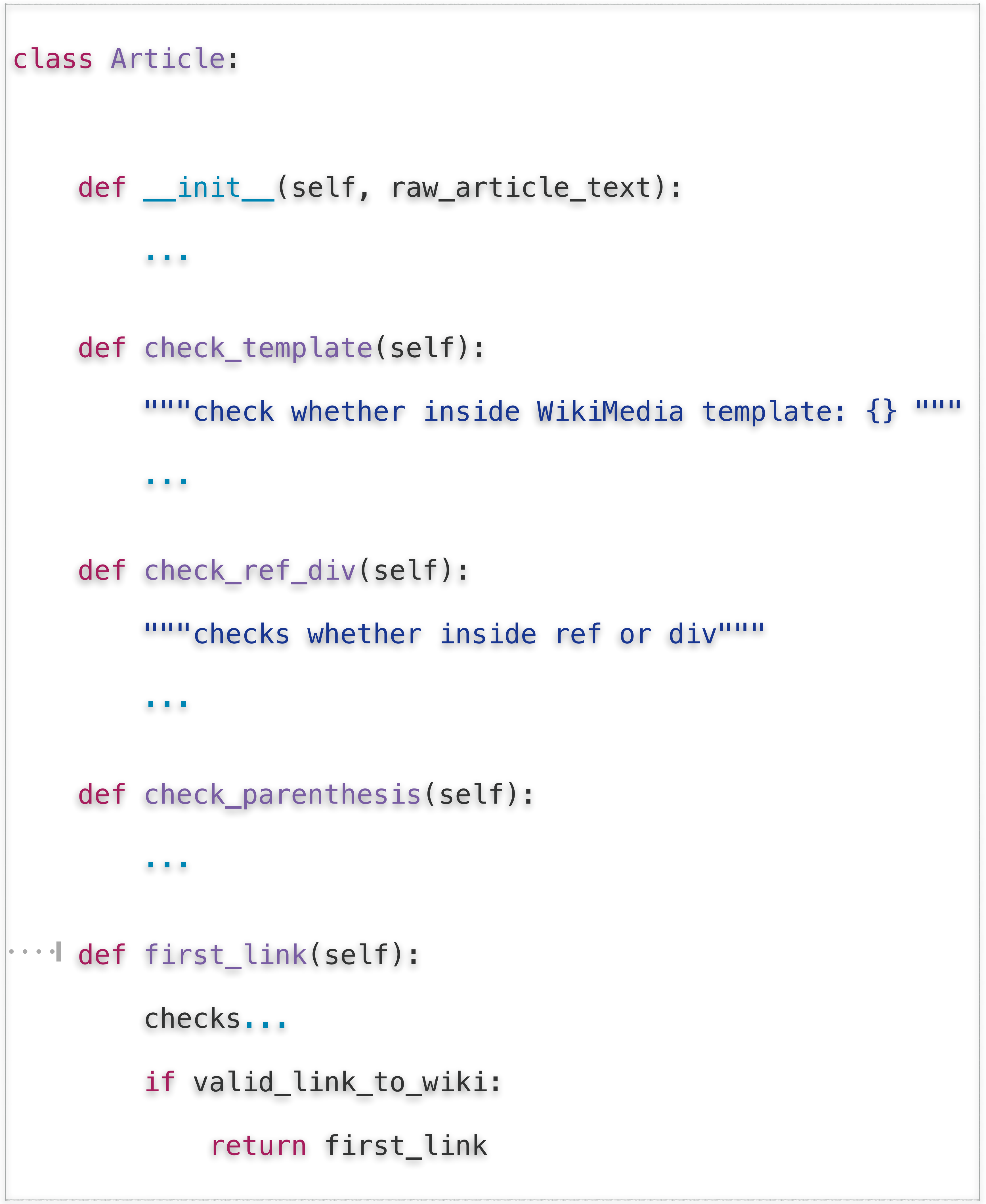


1: inside Wikimedia template?
trigger: {{ }}

2: inside <ref>, <div>?

3: inside ()?

valid link to Wikipedia article? ☑



```
class Article:

    def __init__(self, raw_article_text):
        ...

    def check_template(self):
        """check whether inside WikiMedia template: {} """
        ...

    def check_ref_div(self):
        """checks whether inside ref or div"""
        ...

    def check_parenthesis(self):
        ...

    def first_link(self):
        checks...
        if valid_link_to_wiki:
            return first_link
```