***FOP\_II***

***Project Report***

***(Individual Work)***

***Course Instructor: Dr Saqib Nazir***

**Name: Ahmad Aleem Akhtar**

**CMS ID: 458945**

**Output:**

****

**Note: Download the project zip from github containing all files for output.**

**Project Test Output:**

**A screen shot of a computer program

Description automatically generated**

***Code Explanation:***

The project was divided into 11 problem modules. The following is an explanation of each module and the explanation of the code used to solve each problem:

**Problem 1: Implementing NewsStory Class**

**Key Points:**

* Create a class **NewsStory**.
* Constructor takes **guid**, **title**, **description**, **link**, and **pubdate** as arguments.
* Store these arguments appropriately.
* Implement methods **get\_guid**, **get\_title**, **get\_description**, **get\_link**, and **get\_pubdate** to retrieve the stored informatio

**Problem 2: Implementing PhraseTrigger Abstract Class**

**Key Points:**

* Define a class **PhraseTrigger** as a subclass of **Trigger**.
* Constructor takes a string **phrase** as an argument.
* Implement a method **is\_phrase\_in(text)** which returns True if the entire phrase is present in the text.
* The method should not be case-sensitive.
* Inherit the **evaluate** method from **Trigger**.

evaluate method from Trigger class

**Problem 3: Implementing TitleTrigger Subclass**

**Key Points:**

* Implement a subclass **TitleTrigger** of **PhraseTrigger**.
* Fires when a news item's title contains a given phrase.
* Constructor takes the phrase as an argument.
* Inherit necessary methods from the superclass.

**Problem 4: Implementing DescriptionTrigger Subclass**

**Key Points:**

* Implement a subclass **DescriptionTrigger** of **PhraseTrigger**.
* Fires when a news item's description contains a given phrase.
* Constructor takes the phrase as an argument.
* Inherit necessary methods from the superclass.

**Problem 5: Implementing TimeTrigger Abstract Class**

**Key Points:**

* Define a class **TimeTrigger** as a subclass of **Trigger**.
* Constructor takes time in EST as a string in the format "3 Oct 2023 17:00:10 ".
* Convert time from string to a **datetime** object before saving it.
* Utilize **strptime** and **replace** methods.

**Problem 6: Implementing BeforeTrigger and AfterTrigger Subclasses**

**Key Points:**

* Implement subclasses **BeforeTrigger** and **AfterTrigger** of **TimeTrigger**.
* **BeforeTrigger** fires when a story is published strictly before the trigger’s time.
* **AfterTrigger** fires when a story is published strictly after the trigger’s time.
* Implement **evaluate** method in a couple of lines.

**Problem 7: Implementing NotTrigger**

**Key Points:**

* Implement a class **NotTrigger**.
* Inverts the output of another trigger.
* Constructor takes the other trigger as an argument.
* **evaluate** method returns the negation of the other trigger's **evaluate** method.

)

**Problem 8: Implementing AndTrigger**

**Key Points:**

* Implement a class **AndTrigger**.
* Constructor takes two triggers as arguments.
* Fires only if both inputted triggers would fire on the news story.

.trigger2.evaluate(story)

**Problem 9: Implementing OrTrigger**

**Key Points:**

* Implement a class **OrTrigger**.
* Constructor takes two triggers as arguments.
* Fires if either one or both of its inputted triggers would fire on the news story.

.trigger2.evaluate(story)

**Problem 10: Implementing filter\_stories Function**

**Key Points:**

* Write a function **filter\_stories(stories, triggerlist)** that filters news stories based on triggers.
* Takes a list of news stories and a list of triggers as arguments.
* Returns a list of only the stories for which a trigger fires.

**Problem 11: Implementing read\_trigger\_config Function**

**Key Points:**

* Read trigger configuration from a file (**triggers.txt**).
* Parse the file to create trigger objects based on the specifications.
* Return a list of triggers