Class Exercise:

Processes v1

You are working on a project to analyze a large dataset containing information about online product reviews. The dataset is quite extensive, and processing it sequentially takes a significant amount of time. To speed up the analysis, you decide to utilize multiprocessing in Python.

Here are the requirements for the program:

1. Define a class called **ReviewAnalyzer** that inherits from **multiprocessing.Process**.
2. The **ReviewAnalyzer** class should have an **\_\_init\_\_** method that takes two parameters: **review\_data** (a list) and **process\_id** (an integer).
3. Inside the **\_\_init\_\_** method, store the **review\_data** and **process\_id** in instance variables.
4. Implement the run method in the **ReviewAnalyzer** class. This method should process a subset of the **review\_data** assigned to the process based on its **process\_id.**
5. The processing of the reviews can be a simple task, such as counting the number of positive and negative reviews.
6. Create a list of **ReviewAnalyzer** instances, each with a different **process\_id** and a subset of the **review\_data**.
7. Start each instance of the **ReviewAnalyzer** class using the start method.
8. Join all the processes using the join method to wait for them to finish.
9. Collect and aggregate the results from each process to obtain the final analysis.

Your task is to implement the **ReviewAnalyzer** class and create instances of it to analyze the online product reviews dataset.

Instructions:

import multiprocessing

class ReviewAnalyzer(multiprocessing.Process):

    def \_\_init\_\_(self, review\_data, process\_id):

# Your code here

    def run(self):

# Your code here

def analyze\_sentiment(review):

# Your code here

# Sample review dataset

reviews = [

    "This product is amazing!",

    "The quality is subpar.",

    "Highly recommended.",

    "Not worth the price.",

    "Excellent product!"

]

# Create instances of ReviewAnalyzer

# Your code here

# Join all the processes

# Your code here

# Aggregate the results

# Your code here

Expected output:

Review Analysis Results:

Positive Reviews: 3

Negative Reviews: 2