**REINFORCEMENT LEARNING: Anjana Ramachandran : 21110251**

A news value maximiser: Politically and commercially affiliated media companies are tasked with maximizing the views for certain articles more than others. Build a system that maximises the views for these “aligned” articles.

**Thought Process and Problem Translation**

1. **Problem Definition:**
   In the context of media companies, the problem is to maximize the number of views on articles that are aligned with specific political or commercial agendas. The challenge lies in deciding which articles to promote to the audience, where each article is analogous to a "slot machine" in the K-armed bandit problem.

2. **Key Components:**
   **1. Arms (K-Armed Bandits):** Each arm corresponds to an article that could be promoted. The number of arms, \( K \), is the number of articles available for promotion.

   **2. Rewards:** The reward in this context is the number of views or clicks an article receives. The reward distribution is unknown and needs to be learned over time.

   **3. Exploration vs. Exploitation:** The dilemma is whether to explore new articles to understand their potential for views or to exploit known high-performing articles to maximize views.

   **4. Objective:** The goal is to maximize the cumulative number of views across all articles over time.

3. **Creative Parameters:**
   **1. User Segmentation:** Segmenting users based on their political preferences, geographic location, or past engagement history. This could lead to different reward distributions for the same article depending on the user segment.

   **2. Time Decay:** Articles may lose relevance over time. Introducing a decay factor in the reward distribution could simulate the decreasing popularity of older articles.

   **3. Article Popularity Prediction:** Using a predictive model to estimate the potential reward (views) of an article based on factors like trending topics, keywords, or author popularity.

   **4. External Events:** Consideration of external events like elections, sports tournaments, or economic changes that might temporarily boost the views of related articles.

4. **My Thought Process and Approach:**

**Understanding the Problem**

Imagine you work for a media company that publishes articles online. Your goal is to maximize the number of views your articles get. You have a few different articles (let's say 3), and each article has a different probability of attracting views when it's published. The problem is, you don't know in advance which article will perform the best. So, you need a strategy to figure out which article is the most popular while also making sure you're not missing out on potential views from other articles.

**The K-Armed Bandit Problem**

**K-Armed Bandit:** Imagine a slot machine with multiple arms (K arms). Each arm, when pulled, gives you a reward (views, in our case) with a certain probability. The challenge is to figure out which arm gives the best reward while balancing between trying different arms (exploration) and using the best one found so far (exploitation).

**In our scenario:**

**Articles as Bandits:** Each article you can publish is like an arm of the slot machine. When you "pull" (publish) an article, you get a reward in the form of views.
Exploration: You try out different articles to see how well they perform.
Exploitation: Once you find an article that seems to perform well, you start publishing it more frequently to get more views.

**I have randomly played around with three article “Article A”, “Article B”, “Article C”.**

**And tried to define my set of custom parameters which guides exploration and exploitation’s extent.**

```
company = MediaCompany(K=K, T=T, article_names=article_names)
results, article_choice_log = company.run_stochastic()
```

```
Creating ArticleBandit 'Article A' with p = 0.36
Creating ArticleBandit 'Article B' with p = 0.90
Creating ArticleBandit 'Article C' with p = 0.74
T=0     Article: 'Article C'    Views: 1
T=1     Article: 'Article C'    Views: 0
T=2     Article: 'Article B'    Views: 1
T=3     Article: 'Article C'    Views: 1
T=4     Article: 'Article B'    Views: 1
T=5     Article: 'Article A'    Views: 0
T=6     Article: 'Article B'    Views: 1
T=7     Article: 'Article B'    Views: 1
T=8     Article: 'Article B'    Views: 0
T=9     Article: 'Article A'    Views: 0
T=10    Article: 'Article C'    Views: 1
T=11    Article: 'Article C'    Views: 1
T=12    Article: 'Article C'    Views: 1
T=13    Article: 'Article C'    Views: 1
T=14    Article: 'Article A'    Views: 0
T=15    Article: 'Article A'    Views: 1
T=16    Article: 'Article C'    Views: 1
T=17    Article: 'Article A'    Views: 0
T=18    Article: 'Article A'    Views: 0
T=19    Article: 'Article A'    Views: 1
```

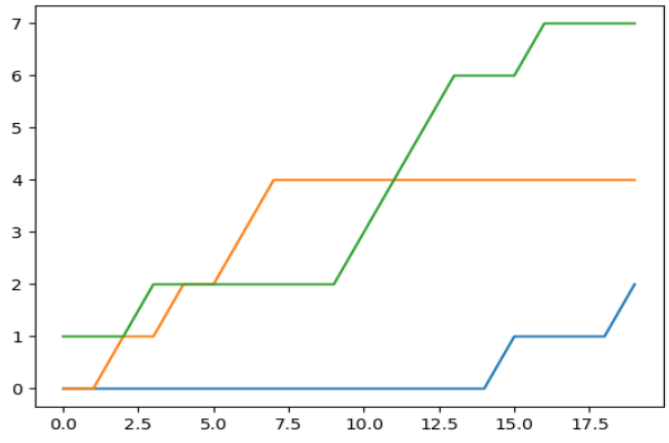
```python
print("\nSummary of Article Performances:")
for article in company.articles:
    print(f"'{article.name}' was published {article.times_pulled} times and got {article.views} views.")

print("\nDetailed Log of Article Choices:")
for log_entry in article_choice_log:
    print(f"Time {log_entry[0]}: Published '{log_entry[1]}', Views: {log_entry[2]}")
```

```
Summary of Article Performances:
'Article A' was published 7 times and got 2 views.
'Article B' was published 5 times and got 4 views.
'Article C' was published 8 times and got 7 views.

Detailed Log of Article Choices:
Time 0: Published 'Article C', Views: 1
Time 1: Published 'Article C', Views: 0
Time 2: Published 'Article B', Views: 1
Time 3: Published 'Article C', Views: 1
Time 4: Published 'Article B', Views: 1
Time 5: Published 'Article A', Views: 0
Time 6: Published 'Article B', Views: 1
Time 7: Published 'Article B', Views: 1
Time 8: Published 'Article B', Views: 0
Time 9: Published 'Article A', Views: 0
Time 10: Published 'Article C', Views: 1
Time 11: Published 'Article C', Views: 1
Time 12: Published 'Article C', Views: 1
Time 13: Published 'Article C', Views: 1
Time 14: Published 'Article A', Views: 0
Time 15: Published 'Article A', Views: 1
Time 16: Published 'Article C', Views: 1
Time 17: Published 'Article A', Views: 0
Time 18: Published 'Article A', Views: 0
Time 19: Published 'Article A', Views: 1
```

```python
cumulative_views = np.cumsum(results, axis=0)
for i, article in enumerate(company.articles):
    plt.plot(cumulative_views[:, i], label=article.name)
```




**This is for the overall cumulative views through the process.**

**I have just involved few parameters. I further want to include parameters like article length, genre of article, targeted audience, and publisher's fame and reliability as additional parameter's to understand the exploration and exploitation further.**