**Algorithms**

**String Algorithm - KMP**

[Assignments]

**1. "The Lost Manuscript"**

Story:

In a mythical land, there existed a fabled manuscript inscribed with ancient wisdom. However, the text was fragmented across multiple stone tablets scattered throughout the kingdom. A young historian, Lucas, discovered a clue suggesting that the tablets, when pieced together, would reveal the hidden manuscript. Each tablet bore a fragment of the manuscript's text.

Task:

Implement the KMP algorithm to help Lucas reconstruct the complete manuscript. Each tablet fragment is represented as a string. Write a function **findCompleteManuscript** that takes in an array of tablet fragments and returns the reconstructed manuscript. If the fragments don't form a complete manuscript, return "Manuscript fragments incomplete."

**2. "The Enchanted Map"**

Story:

In a mystical realm, an ancient map was said to guide the way to a treasure guarded by magical creatures. The map was encoded in a sequence of symbols, but the catch was that only specific patterns within the sequence could unveil the path to the treasure. Adventurer Elara embarked on this quest, equipped with knowledge of the KMP algorithm.

Task:

Elara possesses the encoded map as a string and a set of potential patterns leading to the treasure. Write a function **findTreasurePath** that utilizes KMP to search for the patterns within the map. If any pattern is found, return "Treasure path found," else return "No treasure path found."

**3. "The Secret Code"**

Story:

In a futuristic world, a secret code locked away crucial information vital for the survival of humanity. The code comprised a complex sequence, and cracking it was pivotal. Agents Kai and Maya, skilled in deciphering codes, received the task to decode this sequence using advanced algorithms.

Task:

Create a program that implements the KMP algorithm to decode the secret sequence. The code is provided as a string. Write a function **decodeSecretCode** that takes in the encoded sequence and returns the decoded information. If the code remains indecipherable, return "Code cannot be deciphered."

**4. "The Haunted Cipher"**

Story:

In a haunted mansion, there's a cursed cipher that holds the key to breaking the curse. Legend has it that the cipher, hidden in a sequence of eerie symbols, requires a specific pattern to reveal the incantation needed to dispel the curse. A brave occultist, Aiden, decides to confront the curse.

Task:

Implement a function **breakCurse** using the KMP algorithm. Aiden possesses the encoded cipher as a string and knows the required pattern. The function should return "Curse broken" if the pattern is found in the cipher; otherwise, return "Curse persists."

**5. "The Bard's Song"**

Story:

In a realm known for its rich musical heritage, a legendary bard composed a masterpiece, but the sheet music was lost in time. Generations later, a descendant, Seraphina, discovered fragments of the lost song encoded within ancient scrolls. The melody could only be deciphered if the fragments were arranged in the correct order.

Task:

Write a program that utilizes the KMP algorithm to arrange the fragments in the correct order to reveal the bard's song. Each fragment is represented as a string. Create a function **arrangeMelody** that takes an array of fragments and returns the ordered melody. If the fragments are disordered or incomplete, return "Melody fragments incomplete."

**6. "The Timeless Scroll"**

Story:

In a realm where time stood still, a magical scroll containing the secrets of temporal manipulation was divided into multiple sections by an ancient spell. Only by arranging the sections in the correct sequence could the scroll reveal its wisdom. Scholar Emery, intrigued by the scroll, embarks on a quest to decipher it.

Task:

Emery has collected the scroll's divided sections as strings. Implement a function **decodeTemporalScroll** using the KMP algorithm to arrange the sections correctly. If arranged properly, return the decoded wisdom; otherwise, return "Scroll sections disordered."

**7. "The Astral Glyphs"**

Story:

In a distant galaxy, an ancient prophecy foretold of celestial events encoded within mystical glyphs. These glyphs, scattered across the cosmos, held the key to understanding cosmic phenomena. The astronomer Nova, equipped with knowledge of ancient languages, sought to decipher these glyphs.

Task:

Nova has collected glyphs encoded as strings. Write a function **decodeCelestialGlyphs** that employs the KMP algorithm to arrange the glyphs in the correct sequence, unveiling the prophecy. Return the decoded celestial events or "Glyph sequence incomplete" if the arrangement is incorrect.

**8. "The Elemental Riddle"**

Story:

In a realm governed by elemental forces, a riddle concealed the knowledge to harness these powers. The riddle was encoded within a sequence representing the elements, but only the correct pattern could unlock its secrets. Alchemist Damien, seeking mastery over elements, ventured to solve this puzzle.

Task:

Create a function **unlockElementalRiddle** using the KMP algorithm. Damien possesses the encoded elemental sequence and knows the pattern that could unravel the riddle. If the pattern is found, return "Riddle unlocked"; otherwise, return "Riddle remains a mystery."

**9. "The Dreamer's Code"**

Story:

In a realm where dreams intertwined with reality, a prophetic code revealed glimpses of the future. Dreamweaver Lila, adept at interpreting dreams, stumbled upon a fragmented code within a series of dream sequences. The code foretold of an impending cataclysm, and deciphering it was crucial to prevent disaster.

Task:

Implement a function **decipherDreamCode** that uses the KMP algorithm to arrange the fragmented dream sequences in the correct order. If arranged properly, return the decoded message about the impending cataclysm; otherwise, return "Dream fragments unordered."

**10. "The Labyrinth's Secret Passage"**

Story:

Deep within an ancient labyrinth lay a hidden passage leading to untold treasures. However, the passage was concealed by a sequence of mystical symbols. Explorer Marcus, fascinated by legends of the hidden riches, set out to uncover this secret passage.

Task:

Write a function **findSecretPassage** using the KMP algorithm. Marcus possesses a map with the encoded symbols representing the passage. If the correct pattern leading to the passage is found, return "Secret passage discovered"; otherwise, return "Passage remains hidden."

**11. "The Elemental Keystone"**

Story:

Legends spoke of an elemental keystone, holding the balance of nature's forces. The keystone's location was encrypted within a sequence of elemental symbols. Seeker Aria, entrusted with restoring the harmony, delved into deciphering the sequence.

Task:

Implement a function **findElementalKeystone** using KMP. Aria possesses the encoded sequence and knows the pattern that signifies the keystone's location. Return "Keystone located" if the pattern is found; else, return "Keystone still concealed."

**12. "The Timekeeper's Enigma"**

Story:

In a realm where time was fragile, the Timekeeper left behind an enigma to protect the temporal balance. The enigma was hidden within a sequence representing time fragments. Scholar Remy, driven by the quest for temporal understanding, endeavored to solve the enigma.

Task:

Create a function **solveTimekeepersEnigma** that utilizes KMP. Remy possesses the encoded time fragments. If correctly arranged according to the pattern, return the deciphered enigma; otherwise, return "Enigma remains unsolved."