Class: Transition: - represents a transition

```
def __init__(self, state, symbol):
    """
    Transition constructor
    :param state: state to transit
    :param symbol: symbol
    """
    self.__state = state
    self.__symbol = symbol
```

- getter and setter methods

- a toString method

Class: FiniteAutomata: - represents the finite automata

```
def __init__(self, alphabet, transitions, states, final_states, start_state):
    """
    Constructor for FiniteAutomata class
    :param alphabet: alphabet accepted by the FA
    :param transitions: FA's transitions (state1, state2, s) - means, from state1 go to state2 with the symbol s
    :param states: set of FA's states
    :param final_states: set of FA's final states
    :param start_state: starting state
    """
    self.__alphabet = alphabet
    self.__transitions = transitions
    self.__states = states
    self.__final_states = final_states
    self.__start_state = start_state
```

- getter and setter methods
- a toString method
-

```
def is_seq_accepted(self, sequence):
    """
    Function to check if a sequence is accepted by the FA
    :param sequence: sequence to be verified
    :return: True if the sequence is accepted by the FA, False otherwise
    """
```

```
def longest_prefix_accepted(self, sequence):
    """
    Function to get the longest prefix accepted by a FA
    :param sequence: sequence to be verified
    :return: The longest prefix accepted by the FA
    """
```