MES-System in Python

Arbeitsauftrag:

Entwickle ein rudimentäres Manufacturing Execution System (MES). Es soll in der Lage sein, Produktionslinien zu verwalten, Produktionsaufträge zu erstellen, diese Aufträge zu starten und zu beenden, sowie die Produktionseffizienz zu berechnen.

Die MES-Klasse sollte die folgenden Methoden enthalten:

* add\_production\_line(name): Diese Methode fügt eine Produktionslinie mit dem angegebenen Namen hinzu.
* get\_production\_line(name): Diese Methode gibt eine Produktionslinie mit dem angegebenen Namen zurück.
* create\_production\_order(production\_line\_name, order\_number, product\_name, quantity): Diese Methode erstellt einen neuen Produktionsauftrag auf der angegebenen Produktionslinie mit der angegebenen Bestellnummer, dem Produktname und der Menge.
* start\_production\_order(production\_line\_name, order\_number): Diese Methode startet den angegebenen Produktionsauftrag auf der angegebenen Produktionslinie.
* finish\_production\_order(production\_line\_name, order\_number): Diese Methode beendet den angegebenen Produktionsauftrag auf der angegebenen Produktionslinie.

Es soll eine mes\_utils-Klasse geben, die folgende statische Methoden enthält:

* @staticmethod  
  def get\_order\_by\_number(production\_line, order\_number): Diese Methode findet in der gegebenen Produktionslinie den Auftrag anhand der Auftragsnummer und gibt den Auftrag zurück
* @staticmethod  
  def calculate\_production\_efficiency(order): Diese Methode berechnet die Produktionseffizienz des angegebenen Produktionsauftrags.

Die Programmierung sollte auch eine fehlerhafte Eingabe behandeln, z.B. eine fehlerhafte Produktionslinie oder eine Bestellung mit einer bereits existierenden Bestellnummer. Die MES-Klasse sollte auch die Bestellliste verwalten und es dem Benutzer ermöglichen, eine Bestellung anhand ihrer Bestellnummer abzurufen.

Hinweis: Erstelle für jede Klasse eine eigene Datei.

class ProductionOrder:

    def \_\_init\_\_(self, order\_number, product\_name, quantity):

    def get\_order\_number(self):

    def start(self):

    def finish(self):

    def produce(self, units):

    def get\_production\_efficiency(self):

class ProductionLine:

    def \_\_init\_\_(self, name):

    def add\_order(self, order):

    def get\_production\_line\_name(self):

    def get\_orders(self):

class MES:

    def \_\_init\_\_(self):

    def add\_production\_line(self, name):

    def create\_production\_order(self, production\_line\_name, order\_number:int,

    def produce\_units(self, production\_line\_name, order\_number, units):

    def get\_production\_lines(self):

    def get\_production\_line(self, name):

class mes\_utils:

    @staticmethod

    def get\_order\_by\_number(production\_line, order\_number):

    @staticmethod

    def calculate\_production\_efficiency(order):