**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО**

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт з комп’ютерного практикуму №7

«Розробка моделі на основі петрі-об’єктної технології.»

роботи з дисципліни: « Моделювання систем »

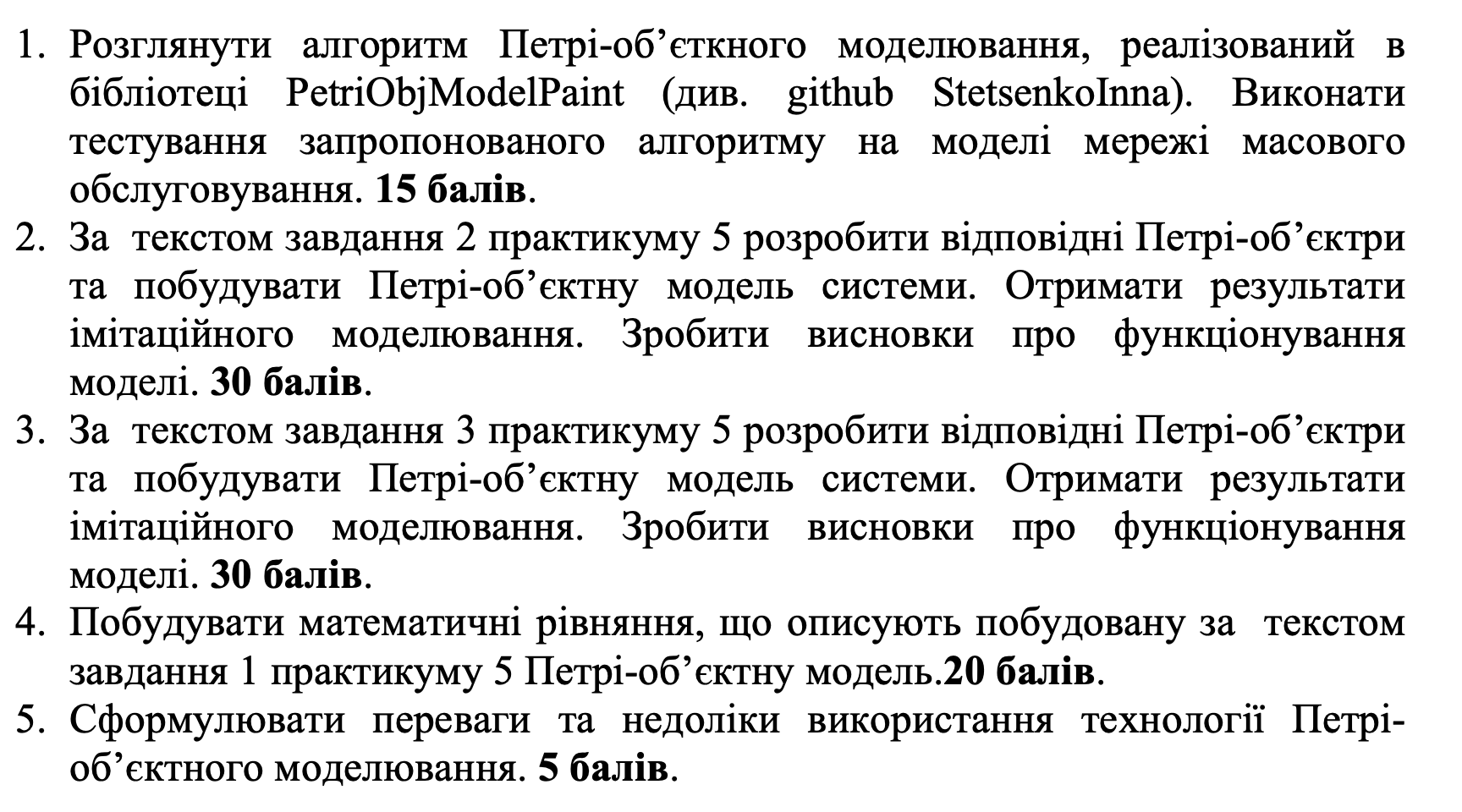
Студент: Мєшков Андрій Ігорович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Група: ІП-15\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Викладач: асистент Дифучин А. Ю.\_\_\_\_\_\_\_\_\_\_

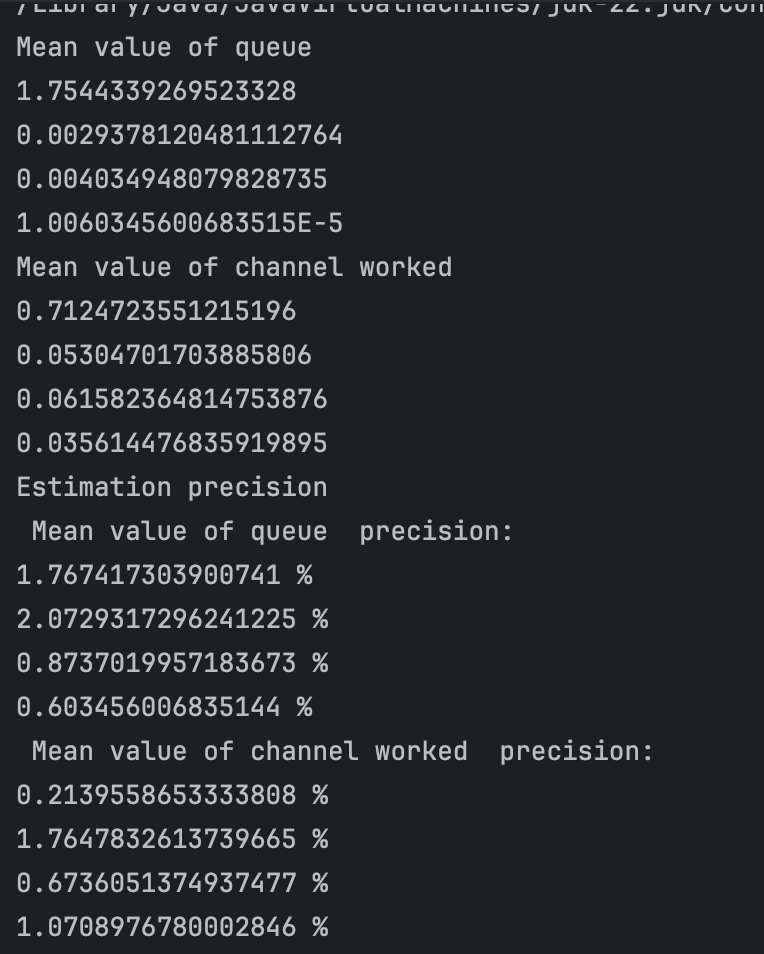
Київ, 2024

# Завдання

****

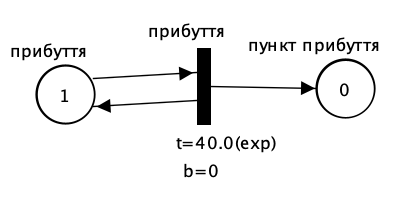
# Хід роботи

Завдання 1. Результати запуску

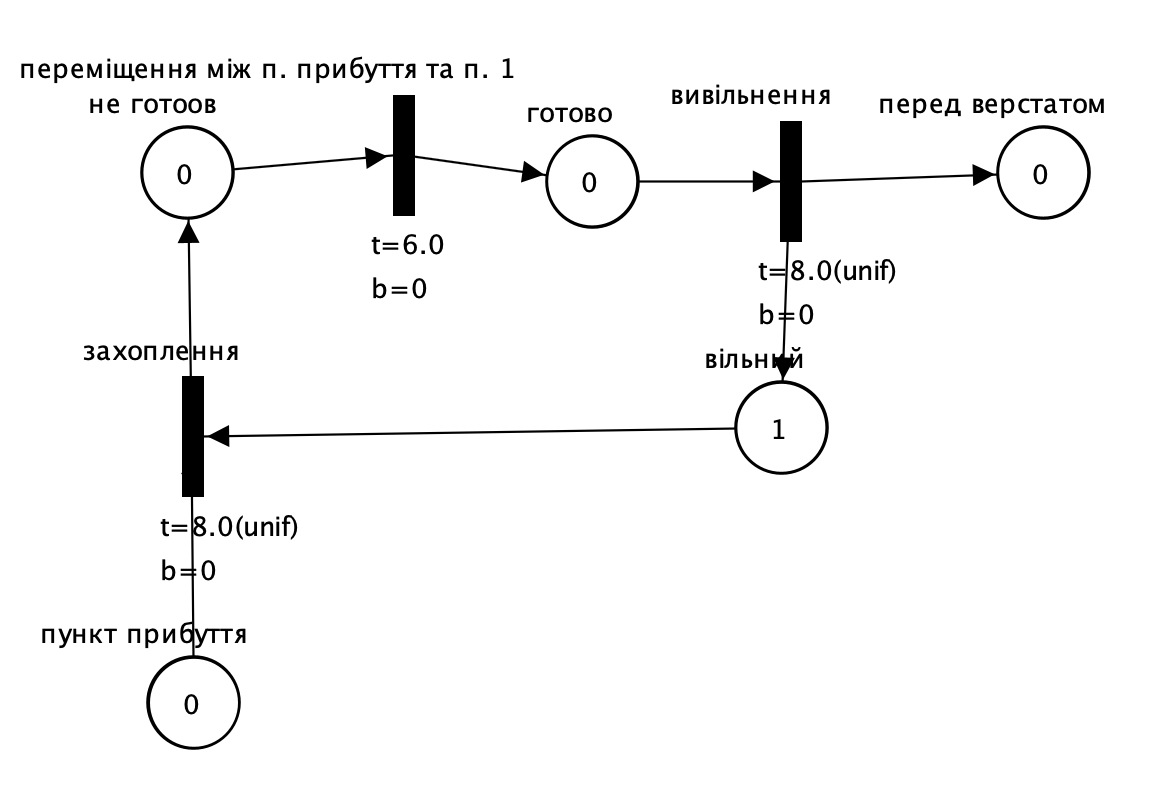


Завдання 2

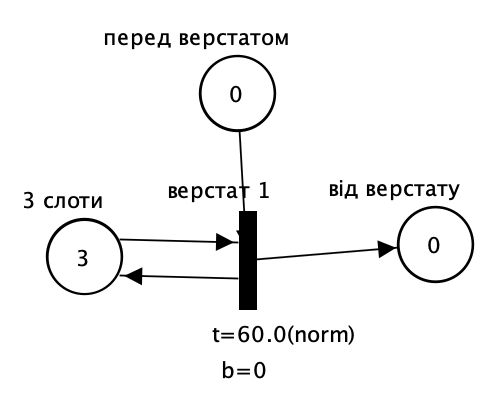
Об’єкт надсилання



Об’єкт перевезення



Об’єкт верстату



Збережемо файли як код та створимо модель.

public static PetriObjModel getModel2() throws ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure{

ArrayList<PetriSim> list = *new* ArrayList<>();

list.add(*new* PetriSim(NetLibrary.Create2()));

list.add(*new* PetriSim(NetLibrary.Transfer2()));

list.add(*new* PetriSim(NetLibrary.Machine2()));

list.add(*new* PetriSim(NetLibrary.Transfer2()));

list.add(*new* PetriSim(NetLibrary.Machine2()));

list.add(*new* PetriSim(NetLibrary.Transfer2()));

list.get(0).getNet().getListP()[1] = list.get(1).getNet().getListP()[0];

list.get(1).getNet().getListP()[3] = list.get(2).getNet().getListP()[0];

list.get(2).getNet().getListP()[2] = list.get(3).getNet().getListP()[0];

list.get(3).getNet().getListP()[3] = list.get(4).getNet().getListP()[0];

list.get(4).getNet().getListP()[2] = list.get(5).getNet().getListP()[0];

PetriObjModel model = *new* PetriObjModel(list);

*return* model;

}

Отримаємо результат на симуляції в 10000

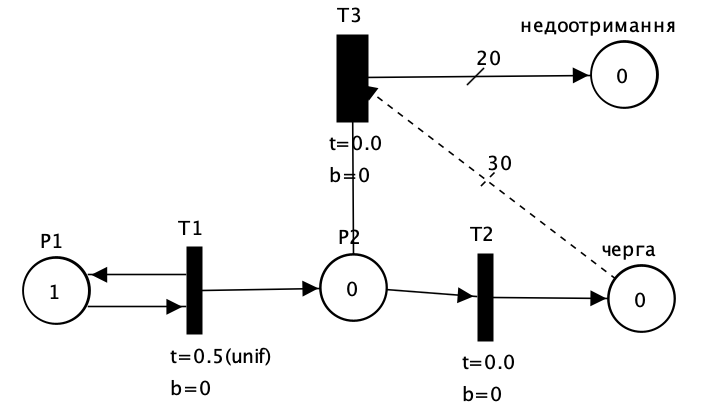


Висновки

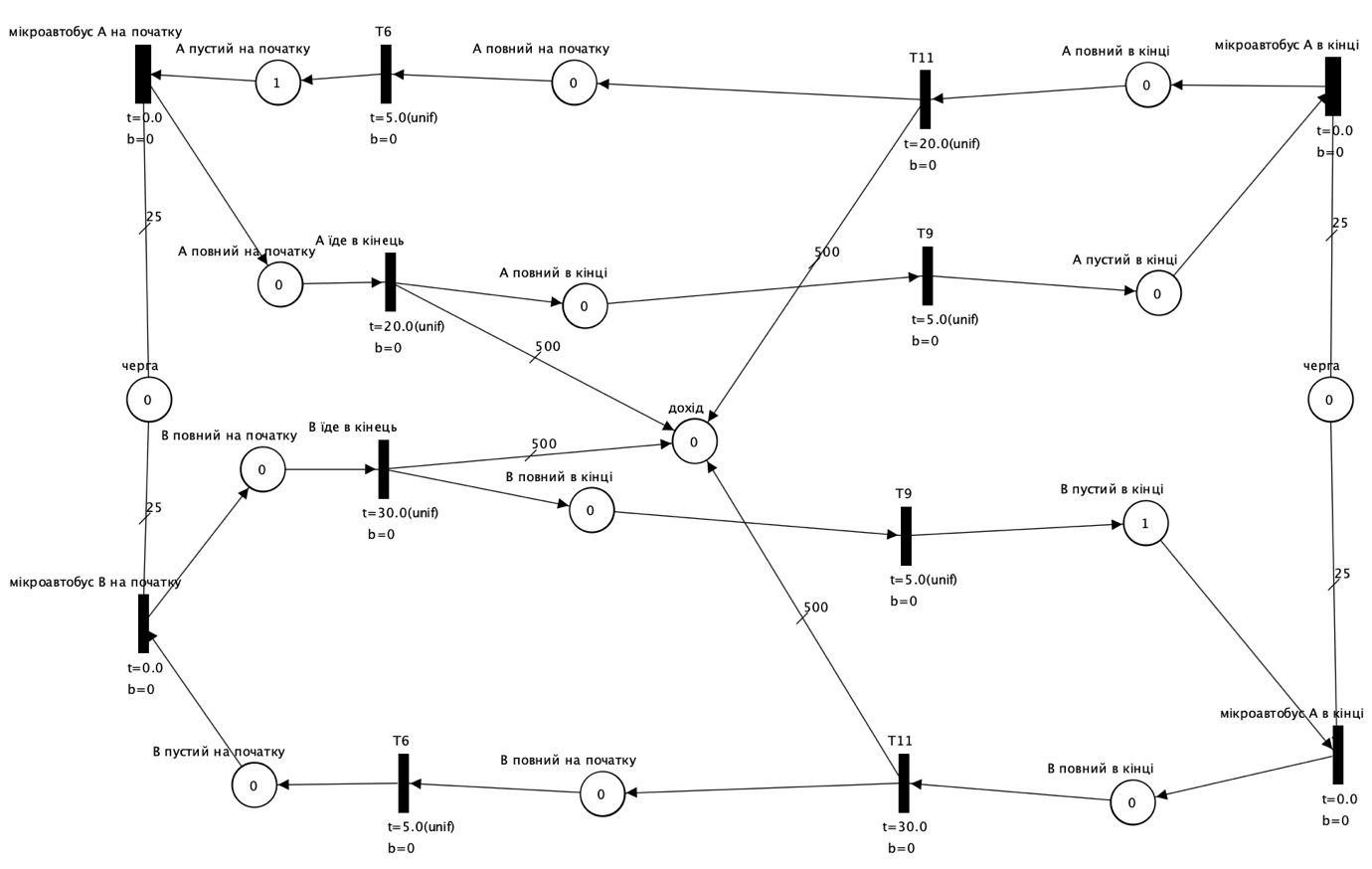
* + Система демонструє нормальне функціонування з послідовним передаванням маркерів через мережу.
  + Однак спостерігається затримка в останньому вузлі, що може бути спричинено низькою продуктивністю або недостатньою швидкістю обробки маркерів в одному з вузлів (Machine2).
  + Слід звернути увагу на оптимізацію параметрів обробки в цих вузлах для уникнення накопичення маркерів.

Завдання 3

Об’єкт черги



Об’єкт доріг



Створимо модель

public static PetriObjModel getModel3() throws ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure{

ArrayList<PetriSim> list = *new* ArrayList<>();

list.add(*new* PetriSim(NetLibrary.Queue3()));

list.add(*new* PetriSim(NetLibrary.Roads3()));

list.add(*new* PetriSim(NetLibrary.Queue3()));

list.get(0).getNet().getListP()[2] = list.get(1).getNet().getListP()[0];

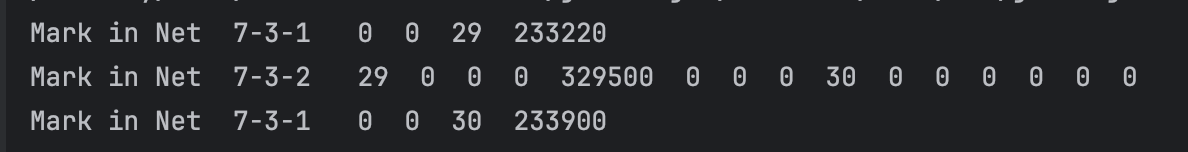
list.get(2).getNet().getListP()[2] = list.get(1).getNet().getListP()[8];

PetriObjModel model = *new* PetriObjModel(list);

*return* model;

}

Отримаємо результат

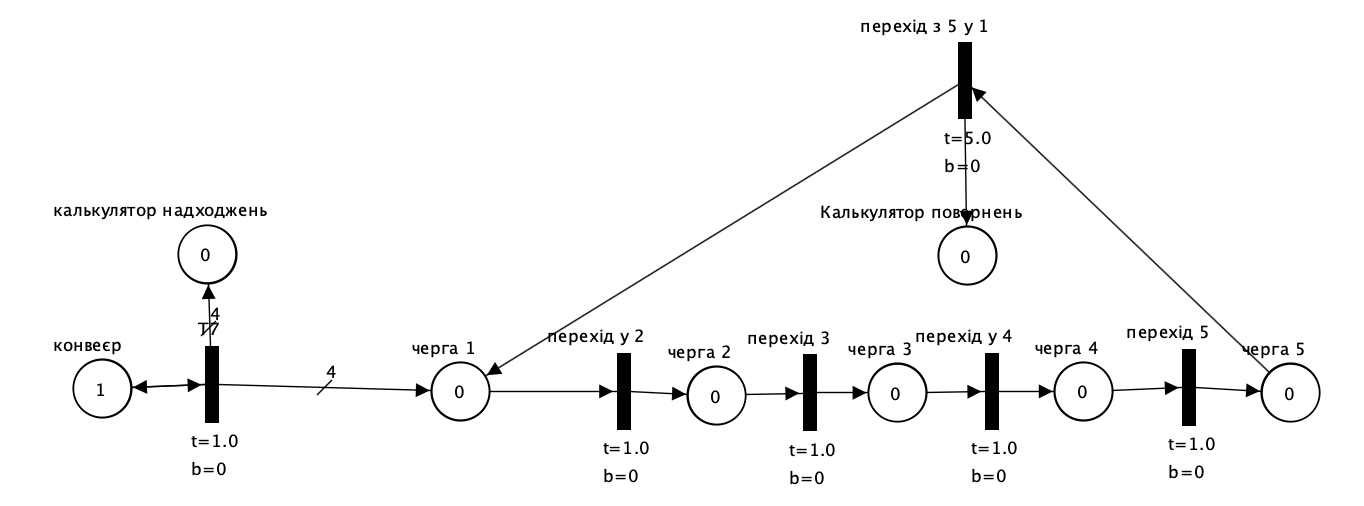


**Висновки**:

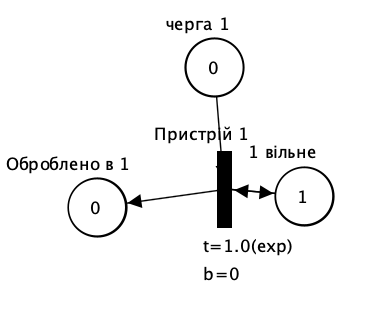
* Основна проблема цієї моделі — перевантаження черг і доріг, що може викликати затримки в обробці маркерів.
* Зменшення кількості маркерів на наступних ітераціях свідчить про те, що система поступово стабілізується, але це займає багато часу.

Завдання 4

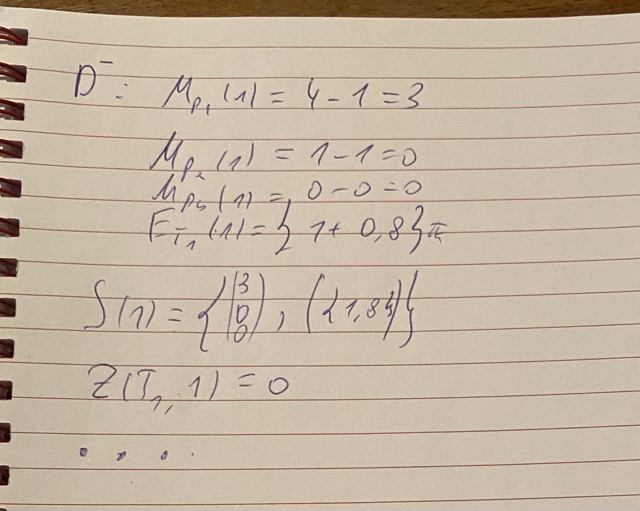
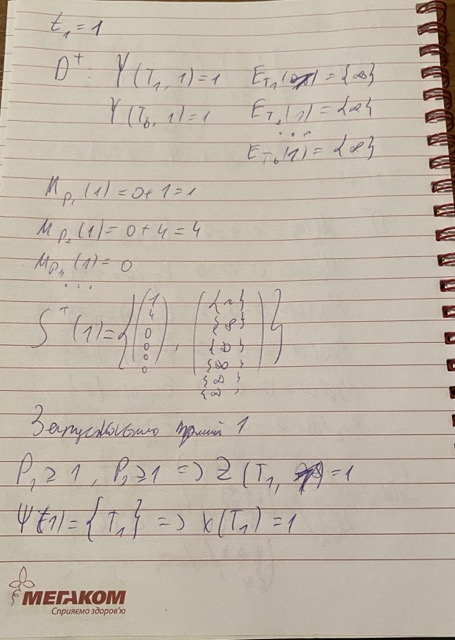
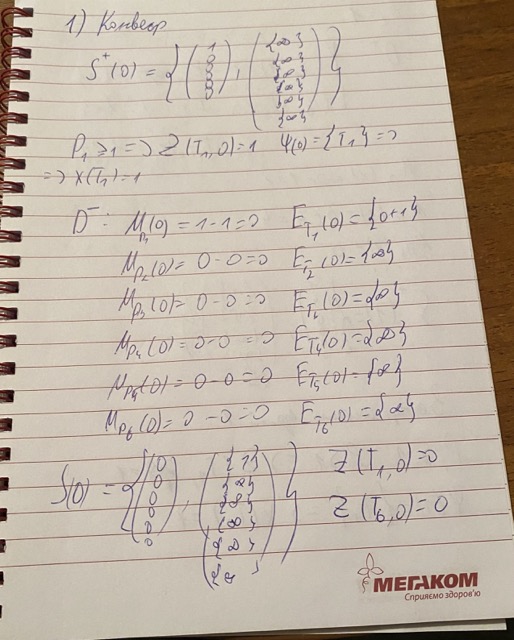
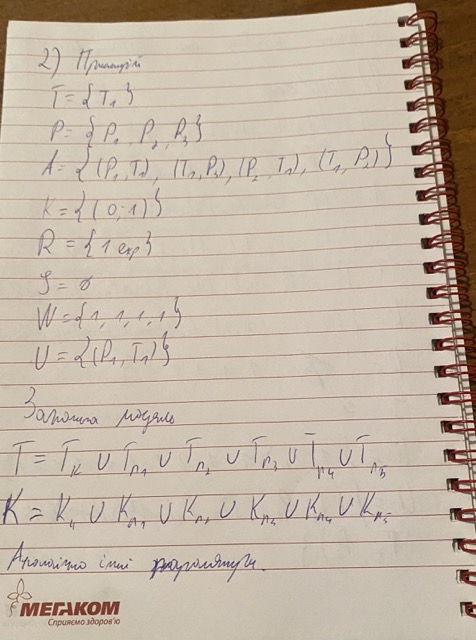
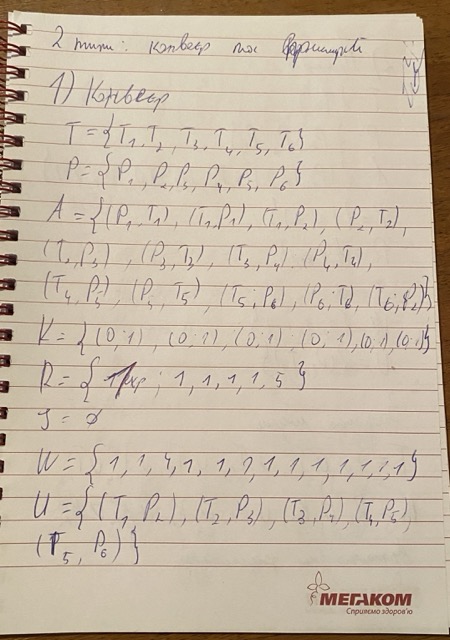
Об’єкт конвеєру



Об’єкт верстату



Побудуємо математичне рівняння



Перевагами Петрі-об’єктного моделювання є універсальність. Оскільки, об’єкти можна перевикористовувати. Також, легше вносити зміни у модель, оскільки не треба повторювати 1 зміну декілька разів.

Недоліками Петрі-об’єктного моделювання є складність. Для малої системи без схожих елементів використання недоцільне.

# ВИСНОВКИ

У результаті виконання практичної роботи було перероблено 3 задачі за допомогою Петрі-об’єктного моделювання. Було побудовано математичне рівняння для однієї моделі.