Екзамен з **“Моделювання систем”**

студента Великого Д.Є., ІС-73, Білет 7

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 126 «Інформаційні системи та технології »
Освітньо-професійна програма «Інформаційні управляючі системи та технології»
Навчальна дисципліна – «Моделювання систем»

ЕКЗАМЕНАЦІЙНИЙ БІЛЕТ № 7

1. Формалізм мереж масового обслуговування.

2. Матричні рівняння базової мережі Петрі.

3. Складіть алгоритм імітації дискретно-подійної системи, що моделює процес обслуговування клієнтів у відділенні банку з шістьма касами протягом робочого дня. Інтервал часу між надходженням клієнтів складає в середньому 0,5 хвилина (за експоненціальним законом розподілу). Час обслуговування клієнтів є випадкова величина з середнім значенням 1,8 хвилини. До кожної каси формується окрема черга. Клієнт, що надійшов на обслуговування, обирає найкоротшу чергу. Метою моделювання є визначення середнього часу обслуговування клієнта в банку.

4. Складіть Петрі-об’єктну модель клієнт-серверного застосування, що обробляє запити на обробку зображень, що надходять з 200 відеокамер спостереження. Інтервал між надходженням пакетів з зображеннями - в середньому 60 хвилин. У кожному пакеті 6 зображень. Тривалість інтелектуальної обробки одного зображення сервером складає в середньому 2 с. Результат обробки зберігається у базі даних протягом 10 мс. Одночасно сервер може обробляти до 100 зображень одночасно. Метою моделювання є визначення величини буфера, який гарантує безвідмовну роботу сервера.

1)

Вимога або заявка або замовлення - запит на обслуговування, який проходить декілька операцій у системі (послідовність операцій є маршрутом запиту, який може бути детермінований або ж ні).

СМО - один або декілька однакових пристроїв обслуговування та одна черга перед ними, одна СМО виконує одну операцію.

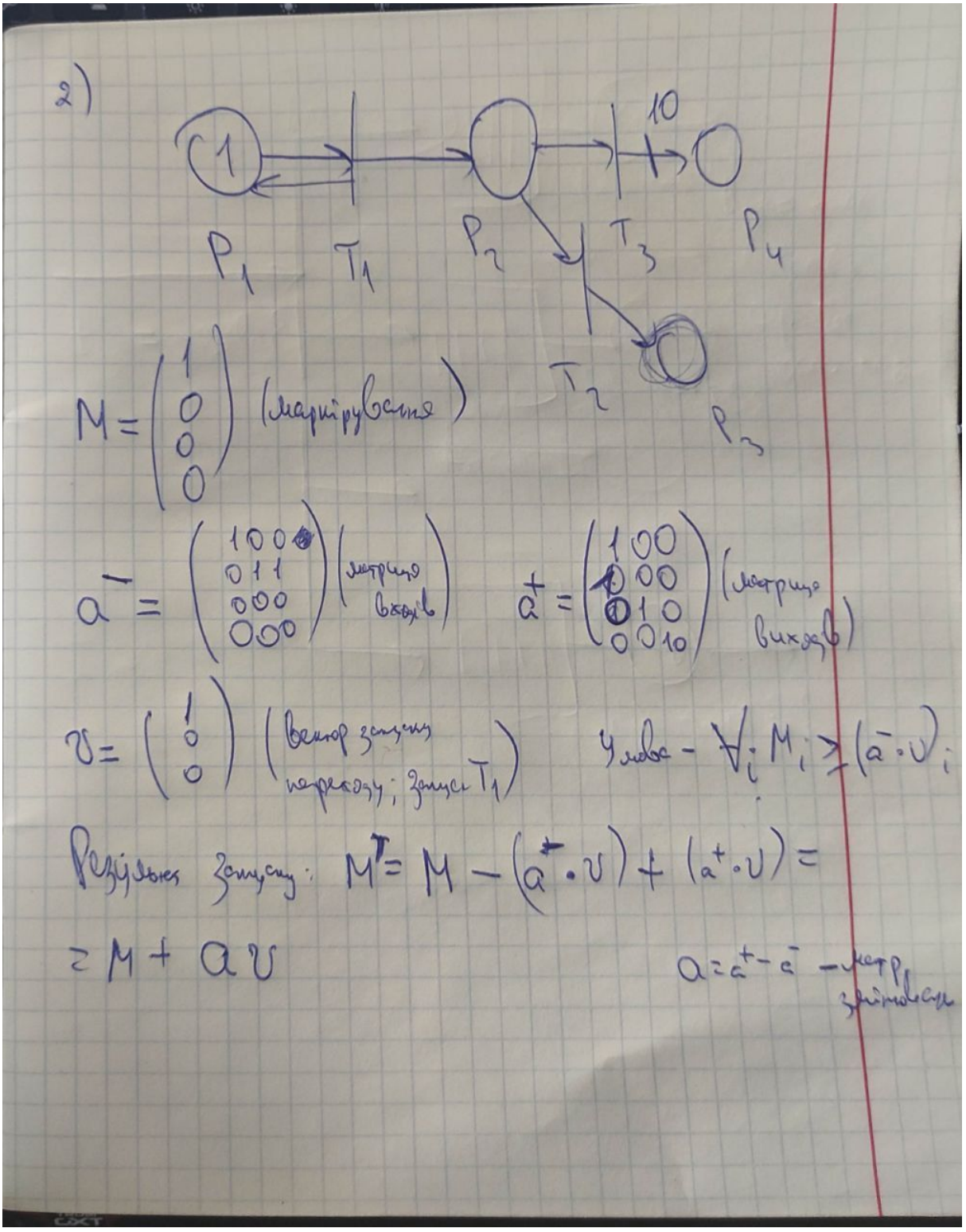
ММО - сукупність СМО із певними маршрутами.

Замкнуті МО - повертаються (проходять ті ж самі операції), а розімкнуті - ні, запити приходять ззовні.

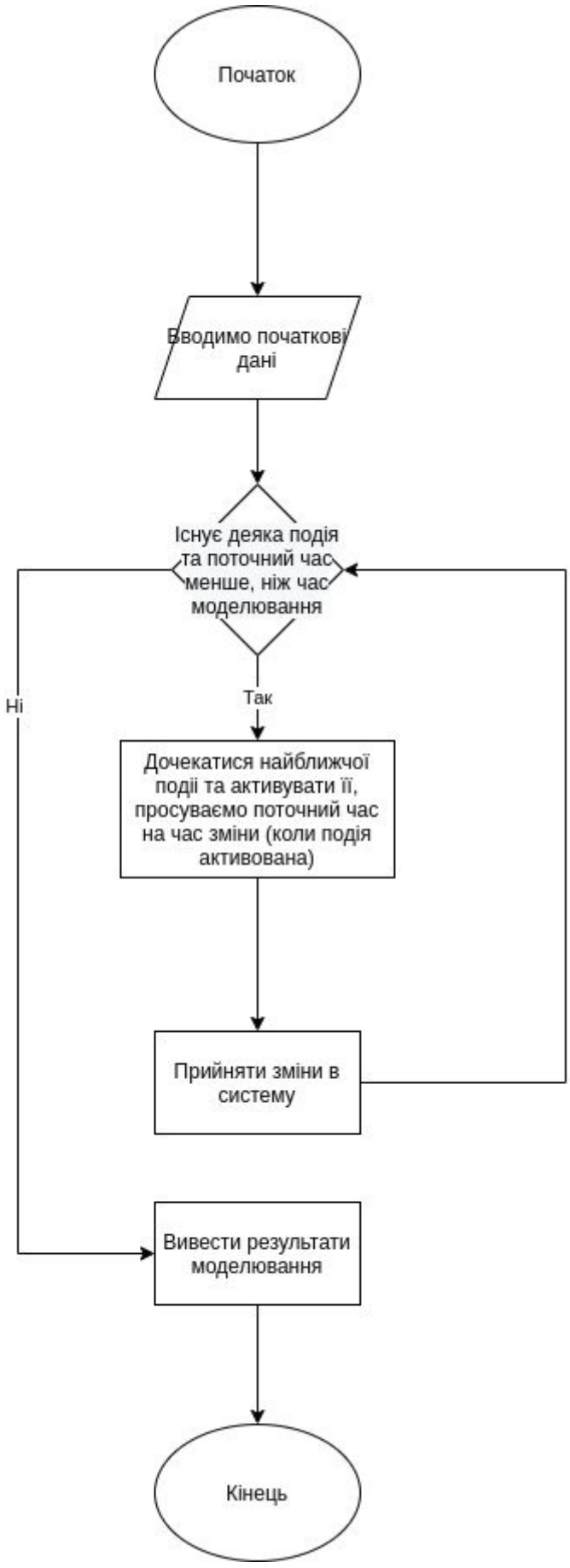
Параметри та вхідні дані ММО - кількість СМО, для кожної СМО - час обслуговування, кількість пристроїв, черга; кількість замовлень, які циркулюють по мережі або параметри інтервали надходжень нового замовлення; параметри маршруту - ймовірності, блокування.

Вихідні дані - ймовірності відмов, середнє завантаження пристроїв, середня довжина черг та середнє очікування в черзі.
Елементи ММО - канал, дуга, черга, багатоканальна СМО (черга з багатьма каналами), розгалуження маршруту, блокування (за вимогою).

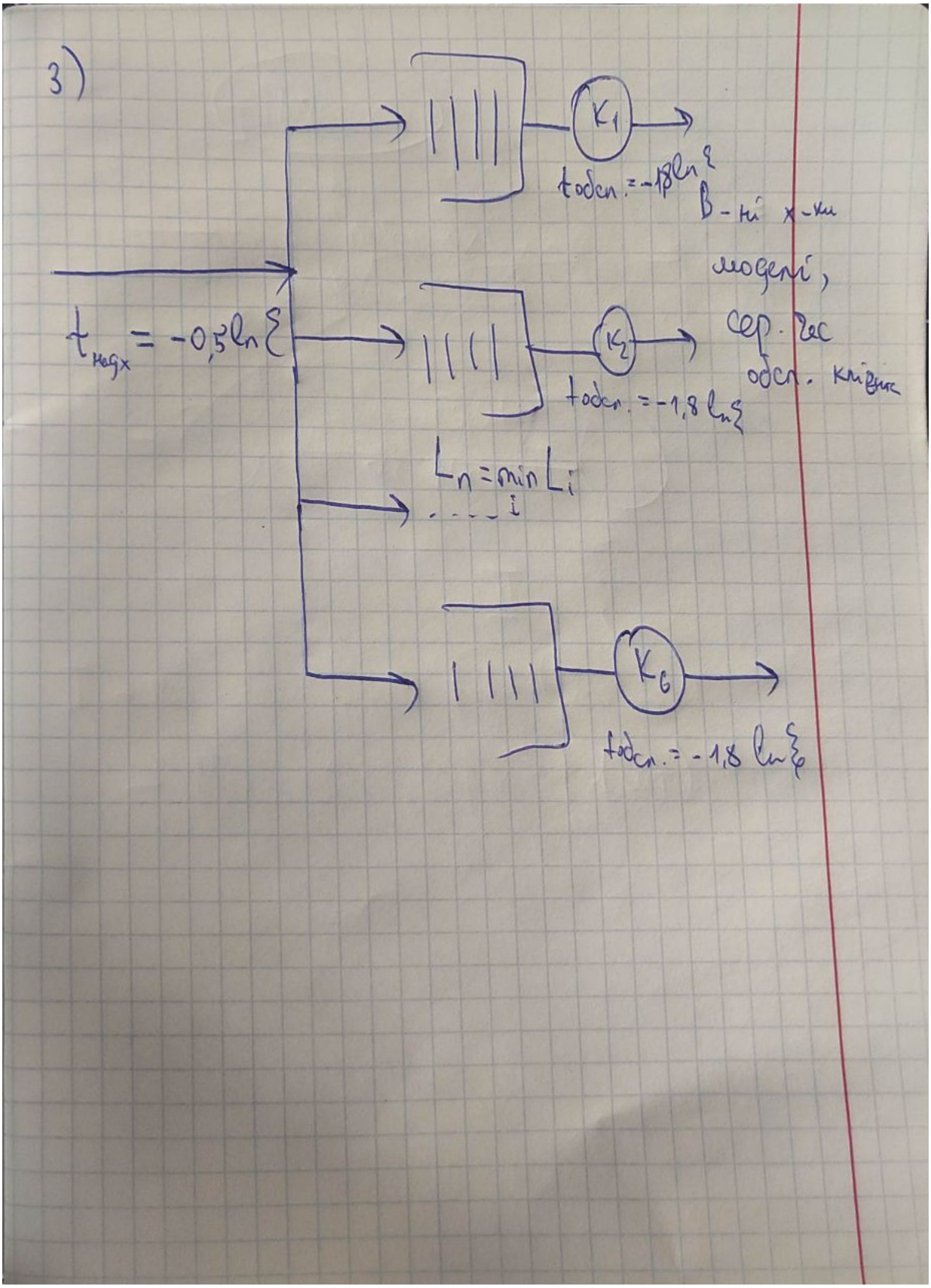
2)



3) Загальний алгоритм дискретно-подійної системи:


Початок
Вводимо початкові дані
Існує деяка подія та поточний час менше, ніж час моделювання
Так
Ні
Дочекатися найближчої події та активувати її, просуваємо поточний час на час зміни (коли подія активована)
Прийняти зміни в систему
Вивести результати моделювання
Кінець

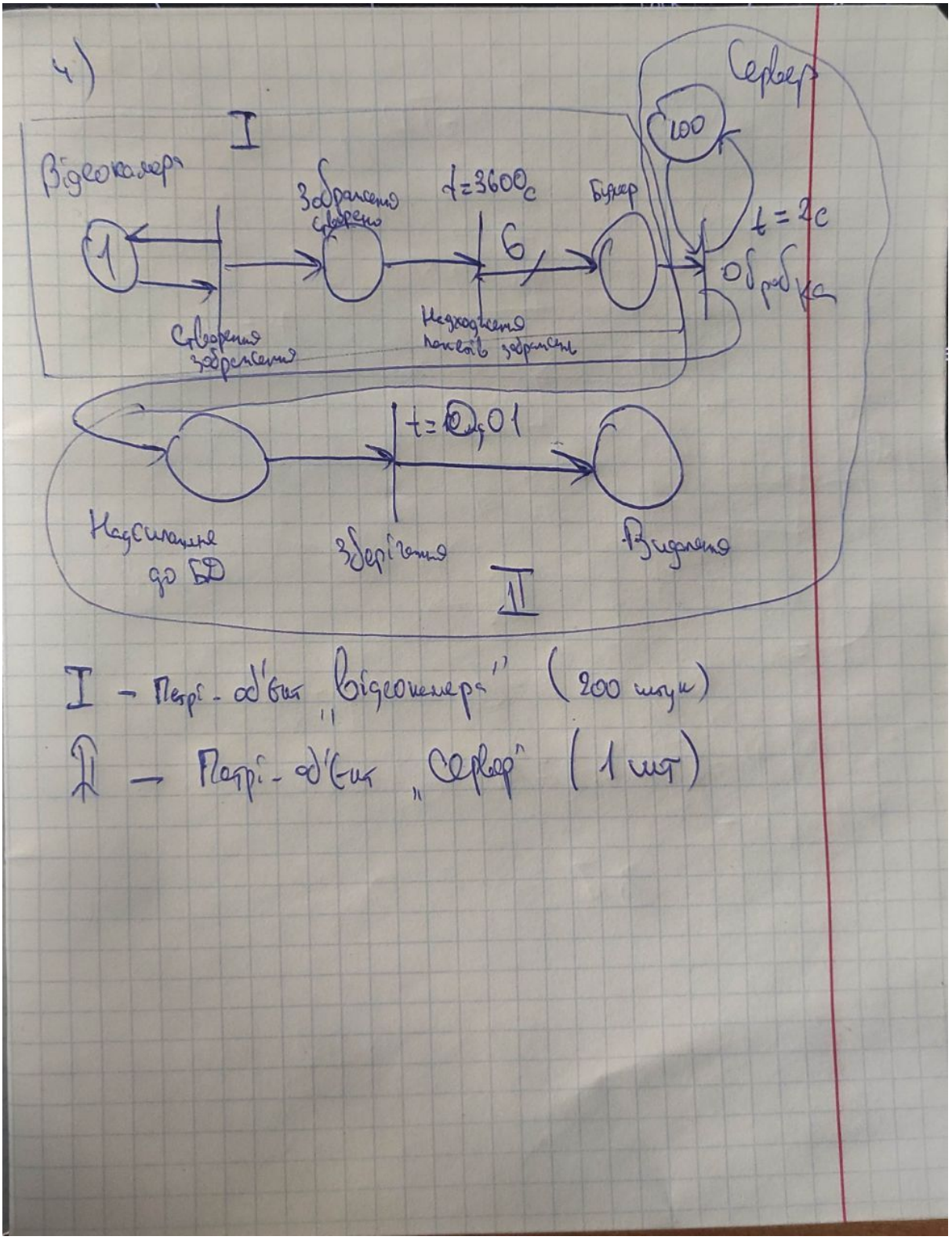
MMO:



Код:

```
import Job from './Job';
import { avg, delay } from './tools';
import _ from 'lodash';
import { once } from 'events';
import Consumer from './Consumer';
import debugHandler from 'debug';

const debug = debugHandler('exam:main');

const constant = { MAX_JOBS_COUNT: 50, MAX_DELAY_TIME: 500, INTERVAL: 10 };

(async () => {
 const consumers = [...Array(6)].map((_, i) => new Consumer(constant, i,
1800));

 const jobs = [...Array(constant.MAX_JOBS_COUNT)].map((_, i) => new Job(i,
constant.MAX_DELAY_TIME));
 _.last(jobs).last = true;

 debug('\nStarting consuming....');
 let jobsNow = [];



 for (const j of jobs) {
   await delay(j.delay);

   const consumer = _.minBy(consumers, 'queueLength');

   await consumer.acceptJob(j);
 }
 await Promise.race(consumers.map((c) => once(c,
Consumer.endConsumingEvent)));
 debug('Ending consuming...\n');

 debug({
   MAX_DELAY_TIME: constant.MAX_DELAY_TIME,
   MAX_JOBS_COUNT: constant.MAX_JOBS_COUNT,
   INTERVAL: constant.INTERVAL,

   averageLoadArr: consumers.map((c) => c.averageLoad),
   averageLoad: avg(consumers.map((c) => c.averageLoad)),
   averageJobsLength: avg(jobsNow),
   averageJobsDoneTime: avg(consumers.map((c) => c.averageJobsDoneInterval)),
   averageTimeWaited: avg(consumers.map((c) => c.averageTimeWaited)),
   averageQueueLengthArr: consumers.map((c) => c.averageQueueLength),
   rejectedJobsPercents: `${( .sum(consumers.map((c) => c.jobsDone.length)) /
constant.MAX_JOBS_COUNT) * 100}%`,
 });
})();
```

4)



I – Перший об'єкт „Відеокамера" (200 штук)

II – Перший об'єкт „Сервер" (1 шт)