Лабораторна робота №4

**Навчальна дисципліна**: Інженерія надійності програмного забезпечення

**Назва**: Визначення надійності програмного забезпечення за результатами тестування та випробувань.

**Мета**: Оволодіти знаннями про визначення надійності програмного забезпечення за результатами тестування та випробувань.

**Завдання**:

1. Вивчити моделі визначення надійності програмного забезпечення.

2. Здобути практичні навички використання моделей для розрахунку надійності програмного забезпечення.

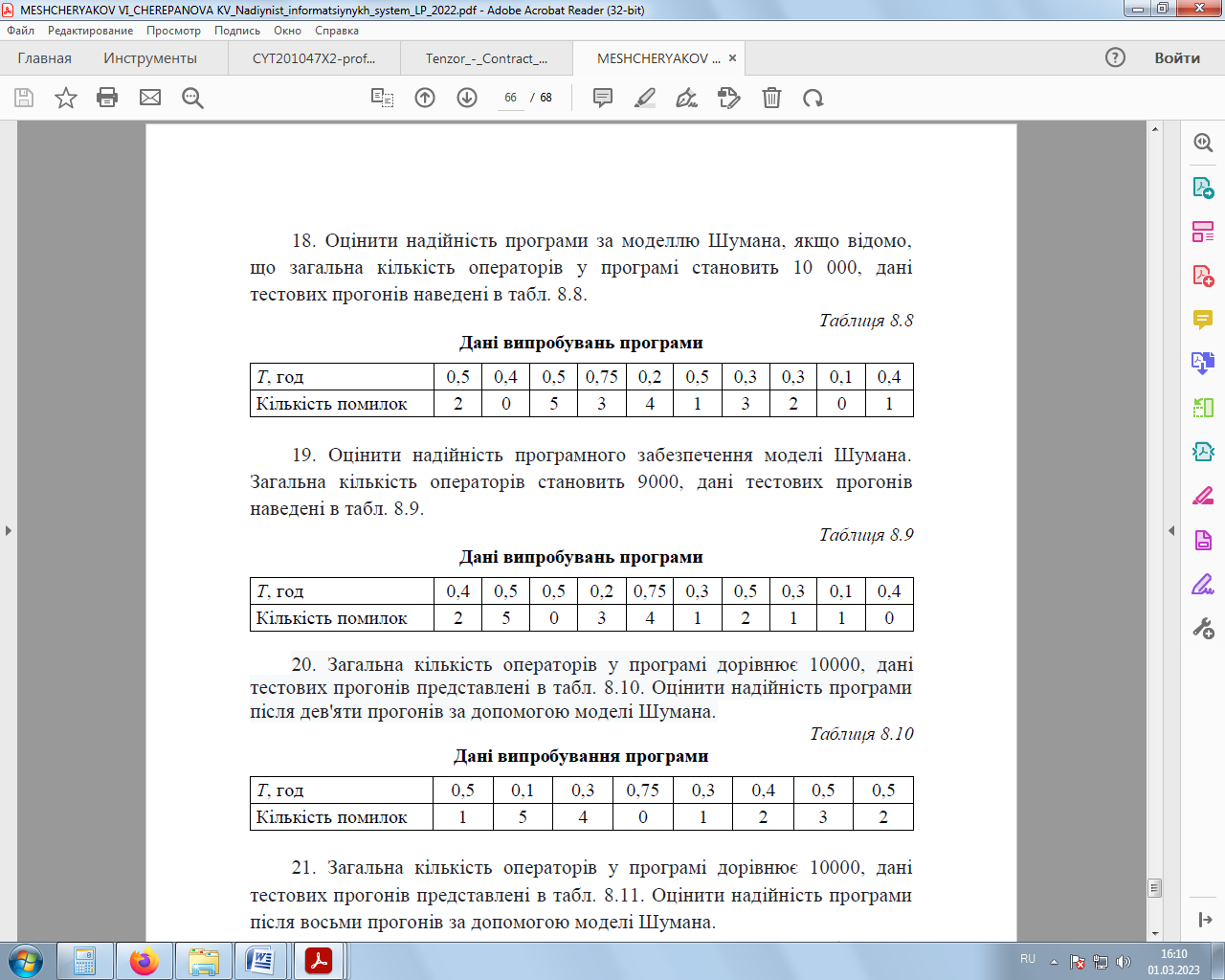
Виконав студент 544 групи

Максимович Микола Юрійович

**Висновки**

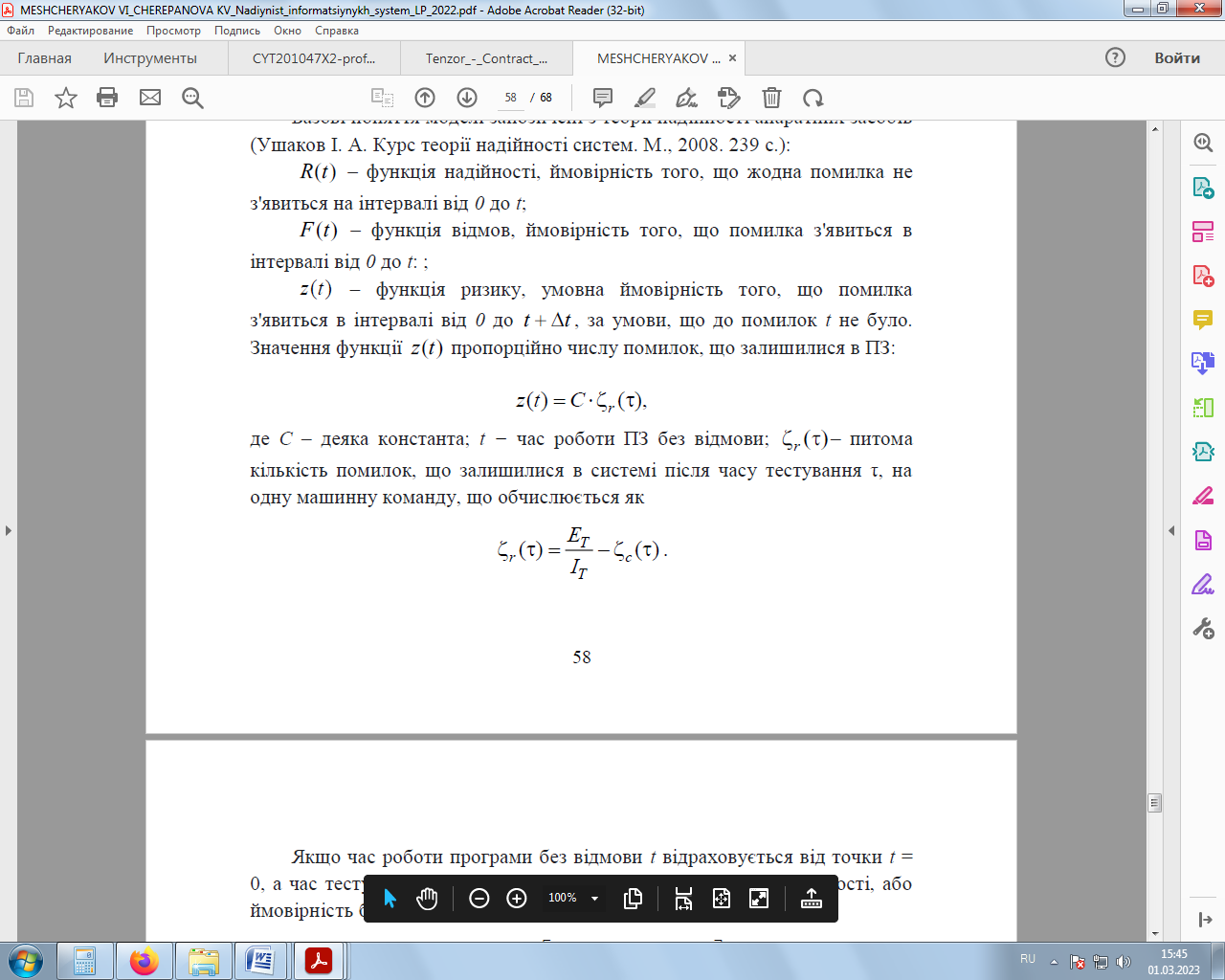
Результатом лабораторної роботи є python скрипт для визначення надійності програмного забезпечення з використанням моделі Шумана.

Вхідними даними в скрипт є таблиця результатів тестувань:

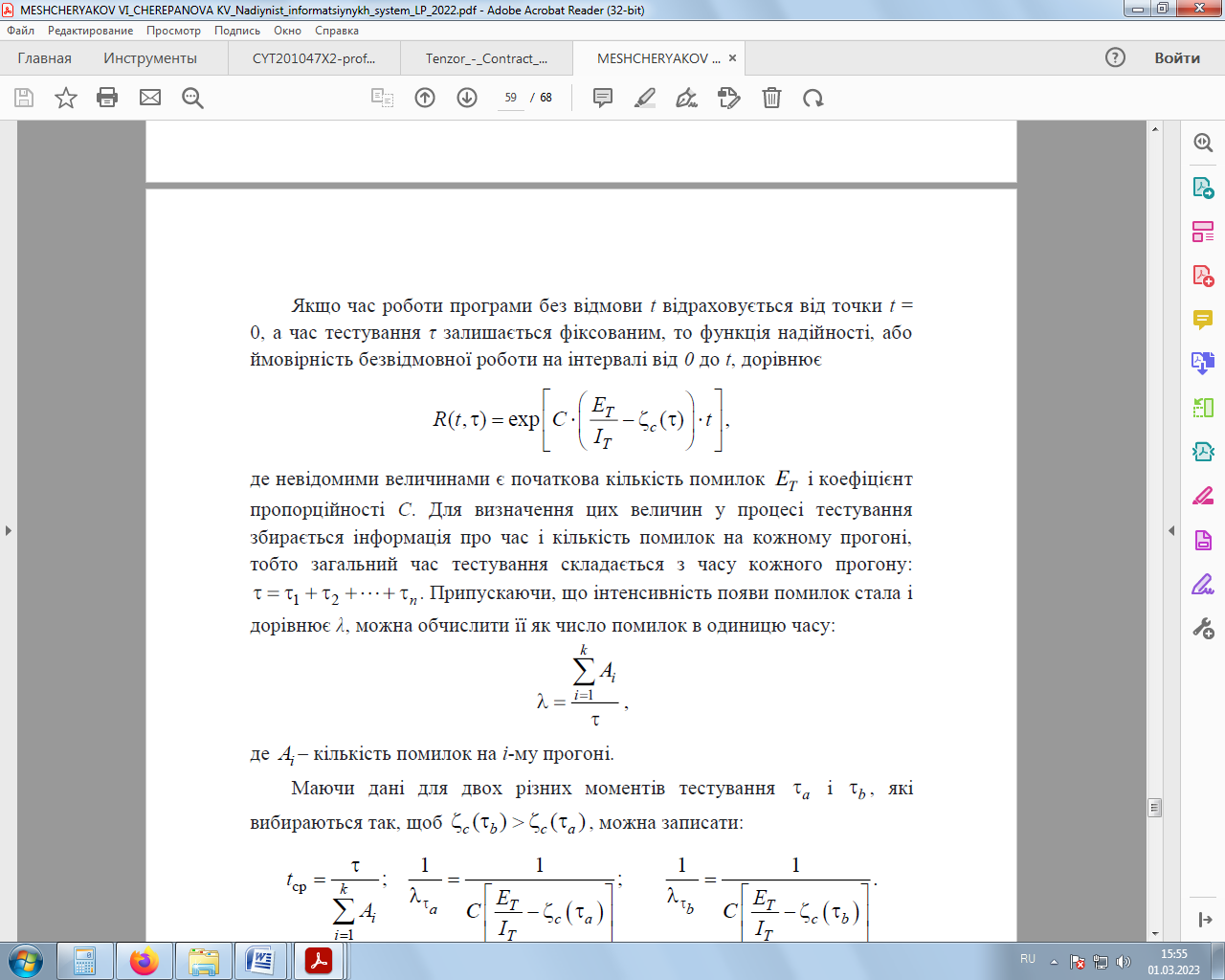


Скрипт працює наступним чином:

Спочатку я визначаю необхідні мені змінні, а часом t я беру значення 0.2 години. Дальше я знаходжу весь час тестування шляхом сумування значень першого рядка. Також знаходжу загальну кількість помилок. Питому кількість помилок, що залишилися в системі після часу тестування τ, на одну машинну команду, я обчислюю за формулою:

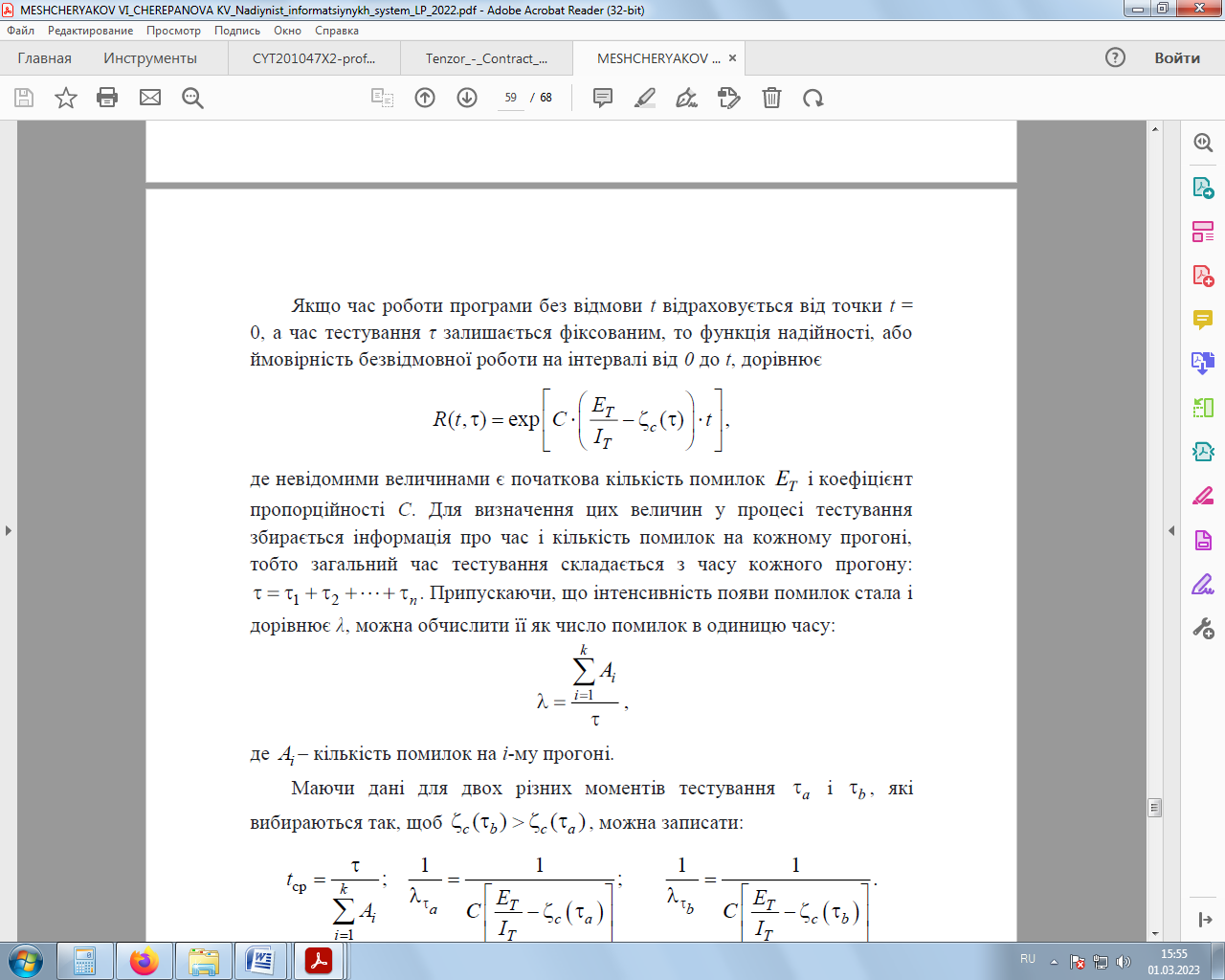


Якщо час роботи програми без відмови *t* відраховується від точки *t = 0*, а час тестування *τ* залишається фіксованим, то функція надійності, або ймовірність безвідмовної роботи на інтервалі від *0* до *t*, дорівнює



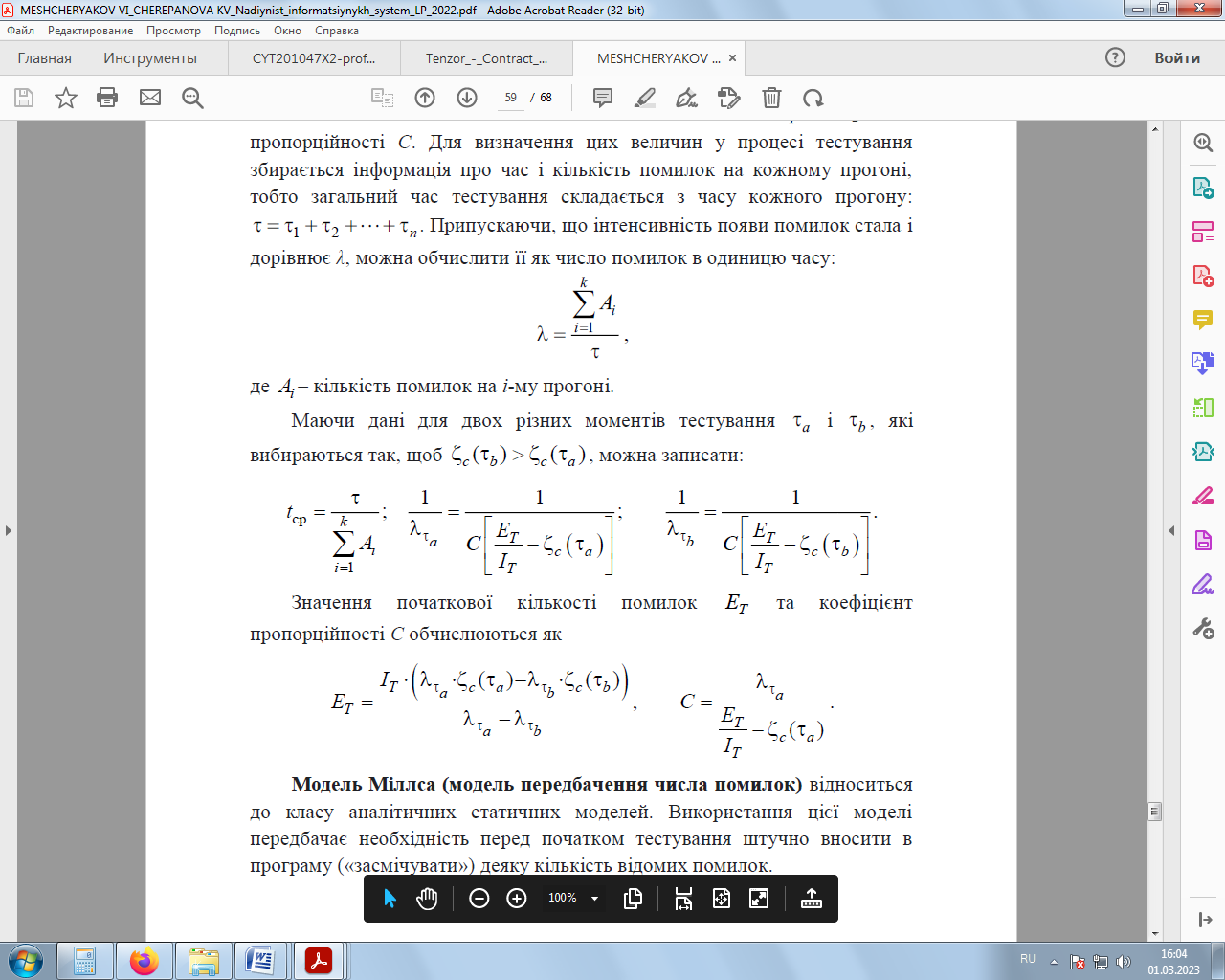
де невідомими величинами є початкова кількість помилок *ЕТ* і коефіцієнт пропорційності *C*.

Припускаючи, що інтенсивність появи помилок стала і дорівнює *λ*, можна обчислити її як число помилок в одиницю часу:

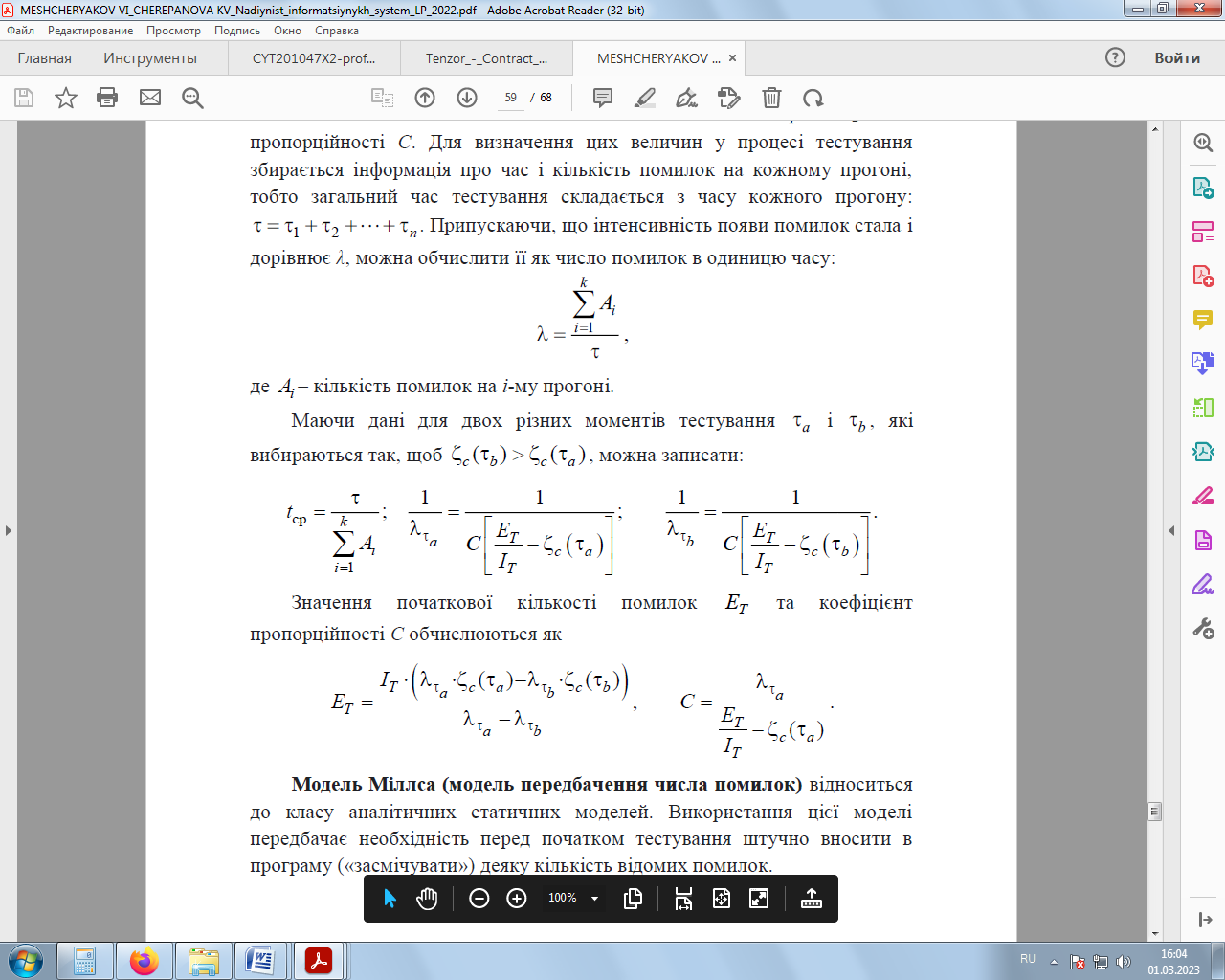


В моєму коді за це значення відповідає змінна y.

І базуючись на формулах наданих в практиці



Значення початкової кількості помилок *ЕТ* та коефіцієнт пропорційності *C* обчислюються як:



Результати виконання коду можна побачити на скріншоті в додатку 2.

**Додатки**

**Додаток 1 – Лістинг коду, файл “main.py”**

import numpy as np

def main():

    arr\_T = [0.5, 0.4, 0.5, 0.75, 0.2, 0.5, 0.3, 0.3, 0.1, 0.4]

    arr\_count\_err = [2, 0, 5, 3, 4, 1, 3, 2, 0, 1]

    T = 0.2

    I\_t = 10000

    a = 3

    b = 4

    r = sum(arr\_T)

    y = sum(arr\_count\_err) / r

    Ecr = sum(arr\_count\_err) / I\_t

    ra = arr\_T[a]

    ya = sum(arr\_count\_err[:a]) / sum(arr\_T[:a])

    Ecra = arr\_count\_err[a] / I\_t

    rb = arr\_T[b]

    yb = sum(arr\_count\_err[:b]) / sum(arr\_T[:b])

    Ecrb = arr\_count\_err[b] / I\_t

    E\_T = (I\_t \* ((ya \* Ecra) - (yb \* Ecrb))) / (ya - yb)

    C = ya / (E\_T/I\_t - Ecra)

    Er = E\_T/I\_t - Ecr

    R\_t\_r = np.exp(C \* Er \* T)

    print("Функція надійності (ймовірність безвідмовної роботи на інтервалі від 0 до",T,"год) =", R\_t\_r)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Додаток 2 – скріншоти результату виконання коду**



Рисунок 1 – результат виконання коду в консолі.

За результатами обчислень додатку, ймовірність безвідмовної роботи є 10.48. Це схоже на правду, адже під час тестування видно що помилки виникають досить часто.

**Контрольні запитання**

1. Моделі надійності програмних засобів можна розділити на дві категорії: детерміністичні моделі надійності (статичні) та імовірнісні моделі надійності. Інша класифікація включає моделі на основі неоднорідного пуассонового процесу, на основі недосконалого відлагодження та компонентні.
2. Аналітичні моделі надійності дають можливість розрахувати кількісні показники надійності, грунтуючись на даних про поведінку програми в процесі тестування. Це означає, що ви можете використовувати ці моделі для аналізу даних з тестування, щоб оцінити, наскільки надійною є ваша програма.
3. Ця модель ґрунтується на наступних припущеннях:

* Константна інтенсивність виявлення помилок: Припущення, що інтенсивність виявлення помилок залишається сталою протягом тестового періоду.
* Незалежність помилок: Помилки в програмному забезпеченні розглядаються як незалежні одна від одної. Це означає, що виявлення однієї помилки не впливає на виявлення інших.
* Постійна інтенсивність роботи системи: Припущення, що програмне забезпечення працює з постійною інтенсивністю протягом тестування.
* Співпадіння часу виявлення та виправлення помилок: Виявлення помилок та їх виправлення відбуваються в той самий час. Це припущення може бути спрощенням, оскільки час виправлення помилок може відрізнятися.

Ці припущення дозволяють побудувати спрощену модель, яка допомагає оцінити надійність програмного забезпечення на основі виявлених помилок та тривалості тестування.

1. Статичні моделі надійності, також відомі як детерміністичні, зазвичай базуються на фіксованих параметрах і не змінюються з часом. З іншого боку, динамічні моделі надійності враховують зміни, які відбуваються з часом, такі як зміни в поведінці програми або в її середовищі.