# Symulacja procesów biologicznych

**Ćwiczenie: 1**
**Autor: Wojciech Laskowski, Bioinformatyka, rok III**

W ramach ćwiczenia zaimplementowano model dynamiczny opisujący zmiany stężenia białek: p53, MDM2 (w formie jądrowej i cytoplazmatycznej) oraz PTEN.

**Wariant: 3**
RK4 zaimplementowany jako funkcja przyjmująca na wejściu dowolny model zapisany za pomocą ODE.

---

| Parametr | Znaczenie | Opis biologiczny |
|---|---|---|
| p1 | produkcja p53 | Stała szybkość produkcji białka p53 w komórce. |
| p2 | produkcja MDM2 | Szybkość transkrypcji MDM2 indukowana przez p53. |
| p3 | produkcja PTEN | Szybkość transkrypcji PTEN aktywowana przez p53. |

---

| Parametr | Znaczenie | Opis biologiczny |
|---|---|---|
| d1 | degradacja p53 | Zależna od poziomu MDM2 w jądrze. Im więcej MDM2, tym silniejsze hamowanie p53. |
| d2 | degradacja MDM2 | Degradacja MDM2 (zarówno jądrowego, jak i cytoplazmatycznego). |
| d3 | degradacja PTEN | Zmniejszenie ilości PTEN z czasem. |

---

| Parametr | Znaczenie | Opis biologiczny |
|---|---|---|
| k1 | transport MDM2 | Kontroluje przepływ MDM2 z cytoplazmy do jądra. |
| k2 | aktywacja transkrypcji | Stała nasycenia dla aktywacji transkrypcji MDM2/PTEN przez p53. |
| k3 | hamowanie transportu MDM2 | Zależność od poziomu PTEN — hamuje migrację MDM2. |

---

| Modyfikacja | Znaczenie | Efekt w modelu |
|---|---|---|
| dna_damage | uszkodzenie DNA | Gdy True, stabilizacja p53 przez obniżenie d2. Gdy False, degradacja MDM2 (a tym samym p53) jest silniejsza. |
| siRNA | terapia siRNA | Gdy True, tłumiona produkcja MDM2 (p2 *= 0.02). |
| pten_active | aktywność PTEN | Gdy False, wyłączona jest produkcja PTEN (p3 = 0). |

---

Dla każdego scenariusza możliwe jest włączenie lub wyłączenie:

- uszkodzeń DNA (dna_damage)
- działania genu PTEN (pten_active)
- działania terapii siRNA (siRNA)

Te wartości dynamicznie modyfikują parametry p2, p3 oraz d2 zgodnie z wytycznymi:

- p2 *= 0.02 przy obecności siRNA (spadek produkcji MDM2)
- p3 = 0 przy nieaktywnym PTEN (brak produkcji)
- d2 *= 0.1 przy braku uszkodzenia DNA (mniejsza degradacja MDM2)

**Każdy scenariusz generuje osobny wykres, przedstawiający czasowy przebieg stężeń białek w logarytmicznej skali, z osią X wyrażoną w minutach (zakres 0–2880 min).**

**Wygenerowane pliki PNG są zapisywane w tym samym folderze co program.**

**Wyniki symulacji i wykresy**

Symulacja została przeprowadzona w czterech scenariuszach:

1. Scenariusz A – Podstawowy:
    - Brak uszkodzeń DNA
    - Aktywny PTEN
    - Brak siRNA

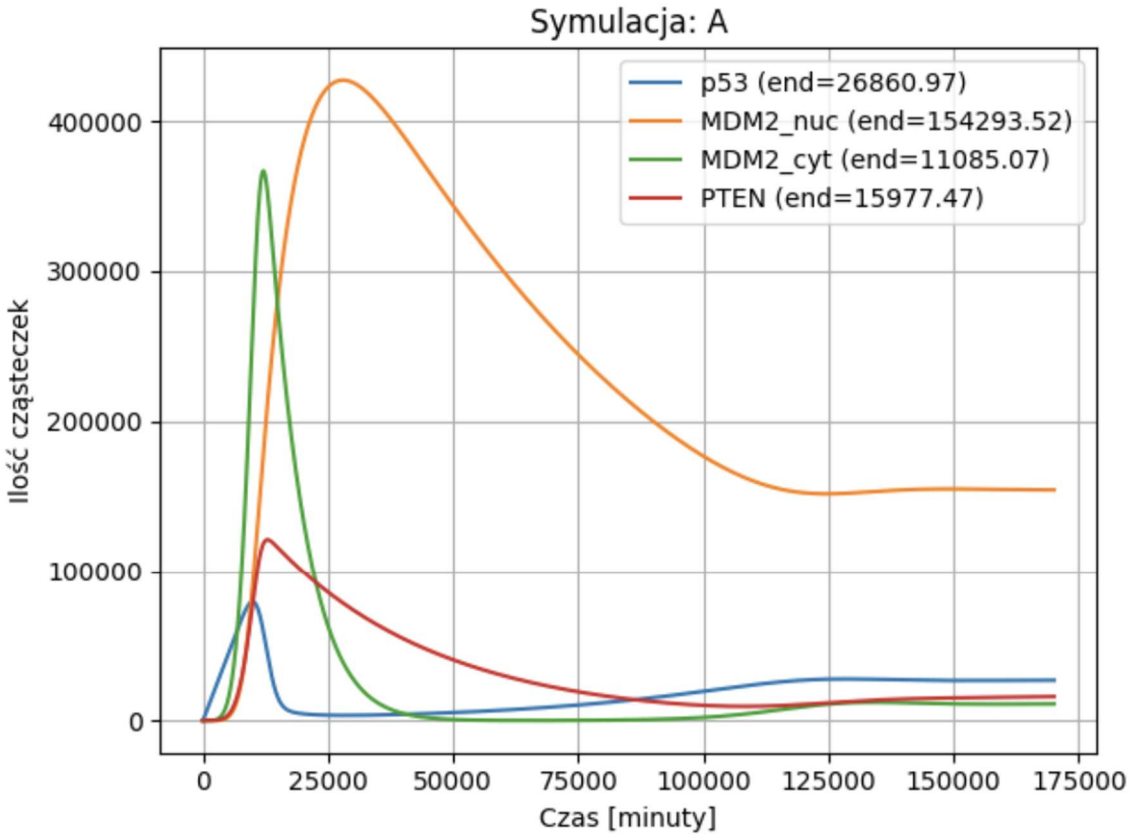
2. Scenariusz B – Uszkodzenie DNA (komórka zdrowa):
    - Uszkodzone DNA
    - Aktywny PTEN
    - Brak siRNA

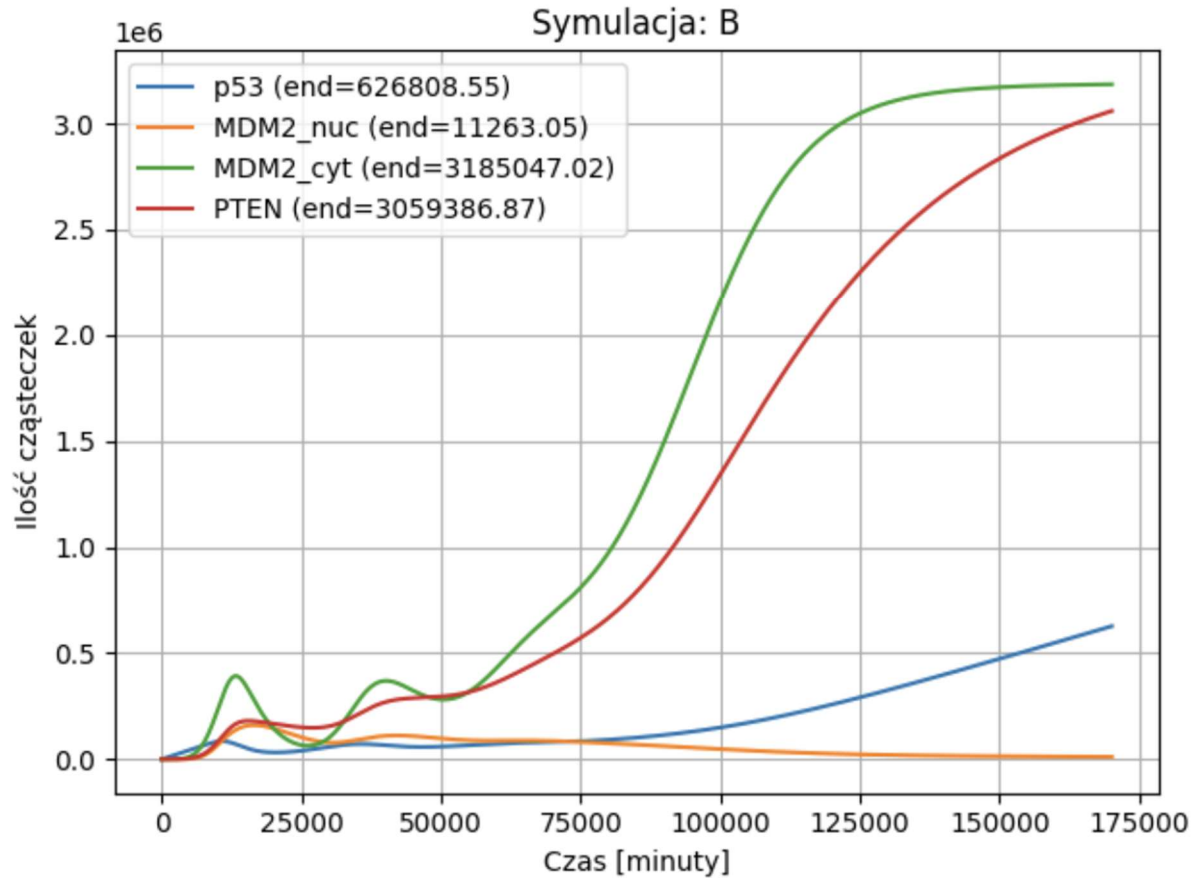
3. Scenariusz C – Komórka nowotworowa:
    - Uszkodzone DNA
    - Nieaktywny PTEN
    - Brak siRNA

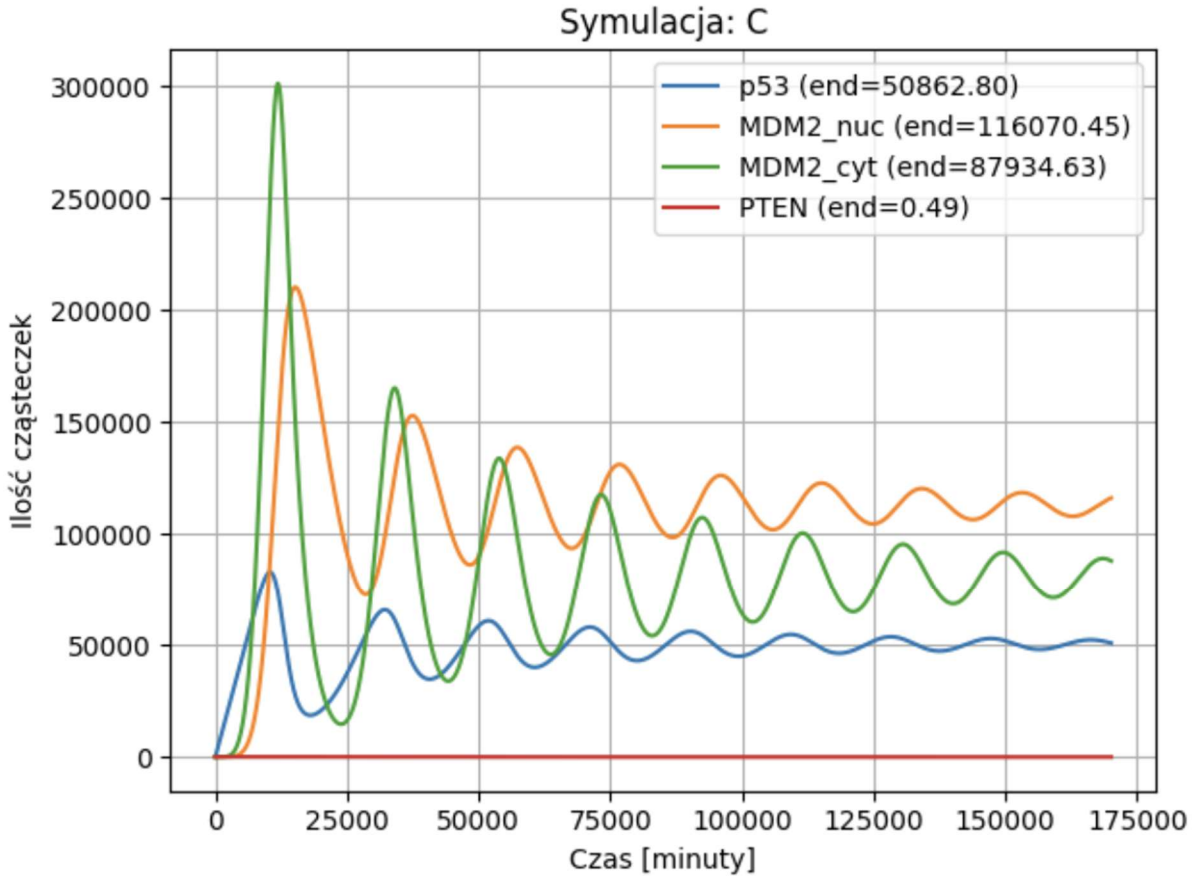
4. Scenariusz D – Terapia:
    - Uszkodzone DNA
    - Nieaktywny PTEN
    - Terapia siRNA włączona

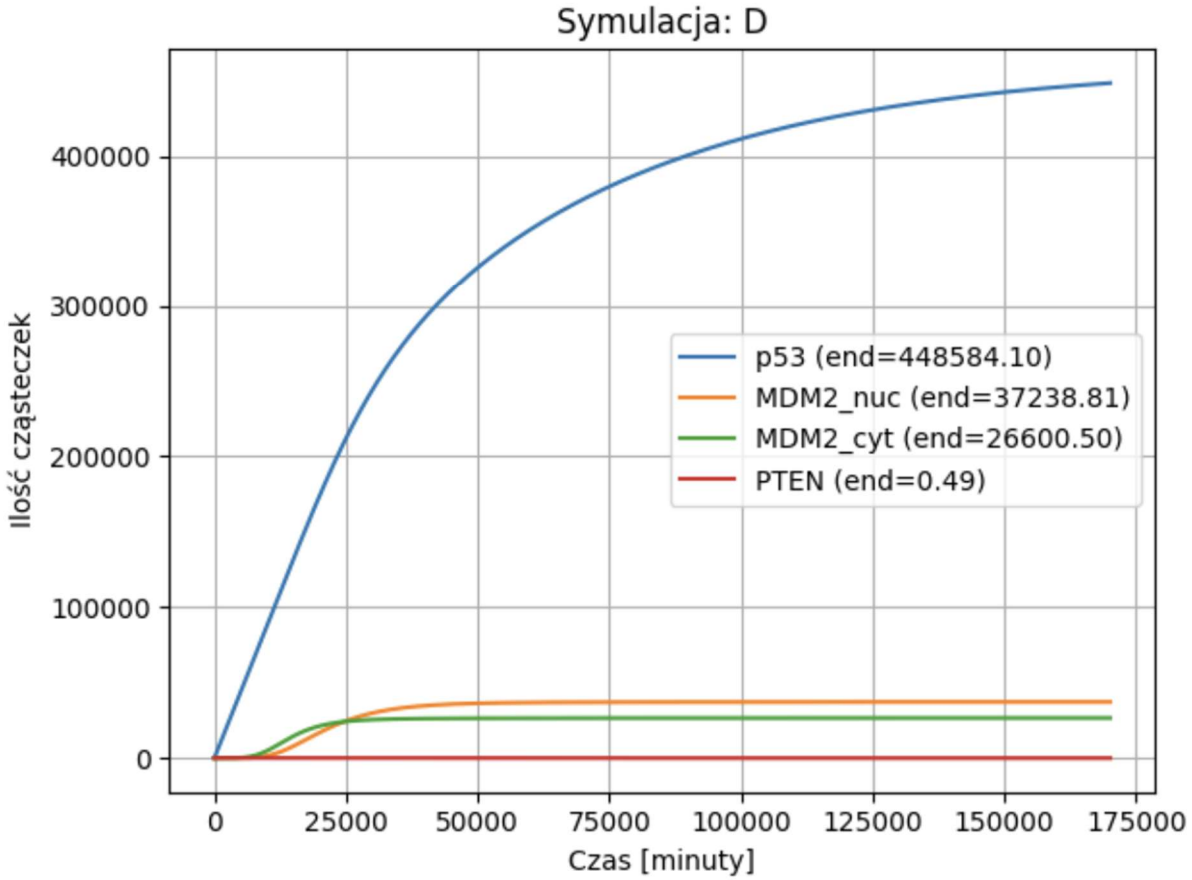
**Symulacje puszczone dla bardzo długiego czasu – wyraźnie widać zmiany w czasie i reakcję komórek**

---

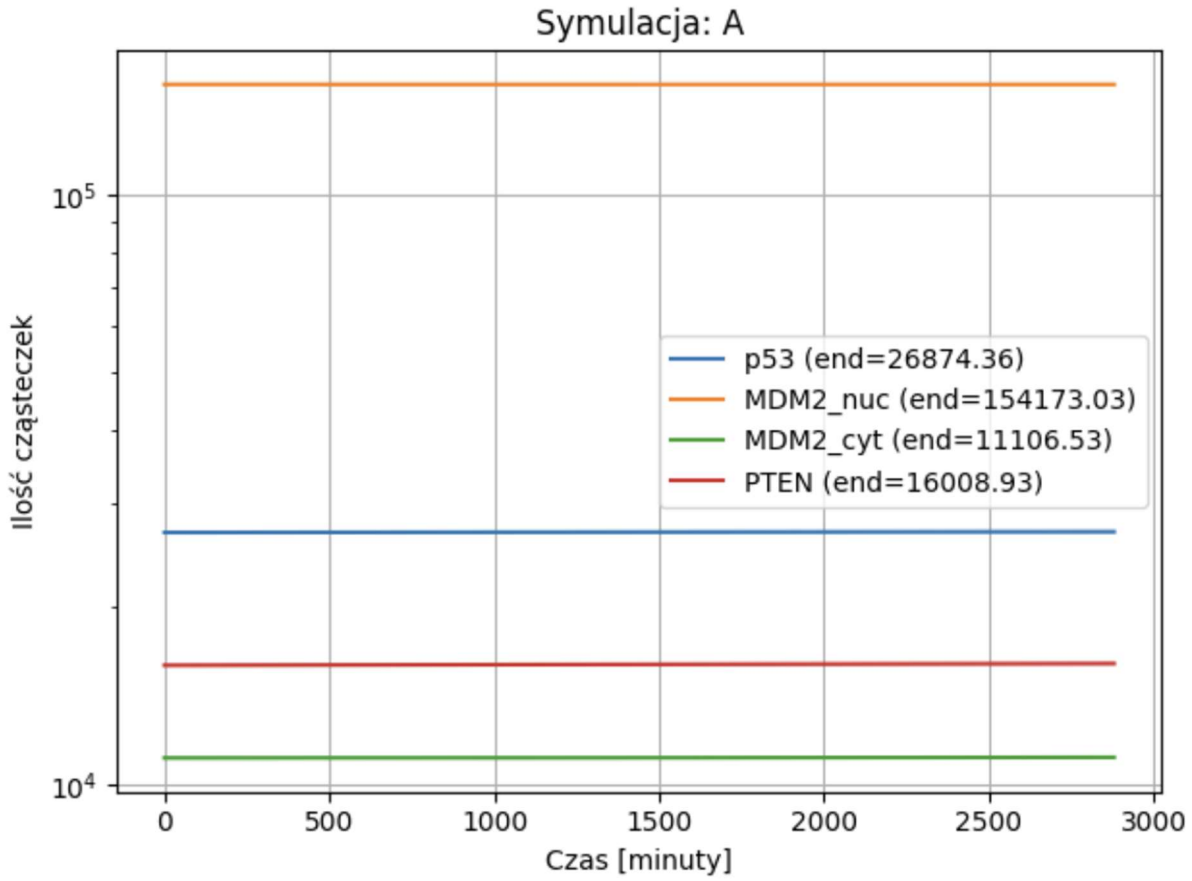


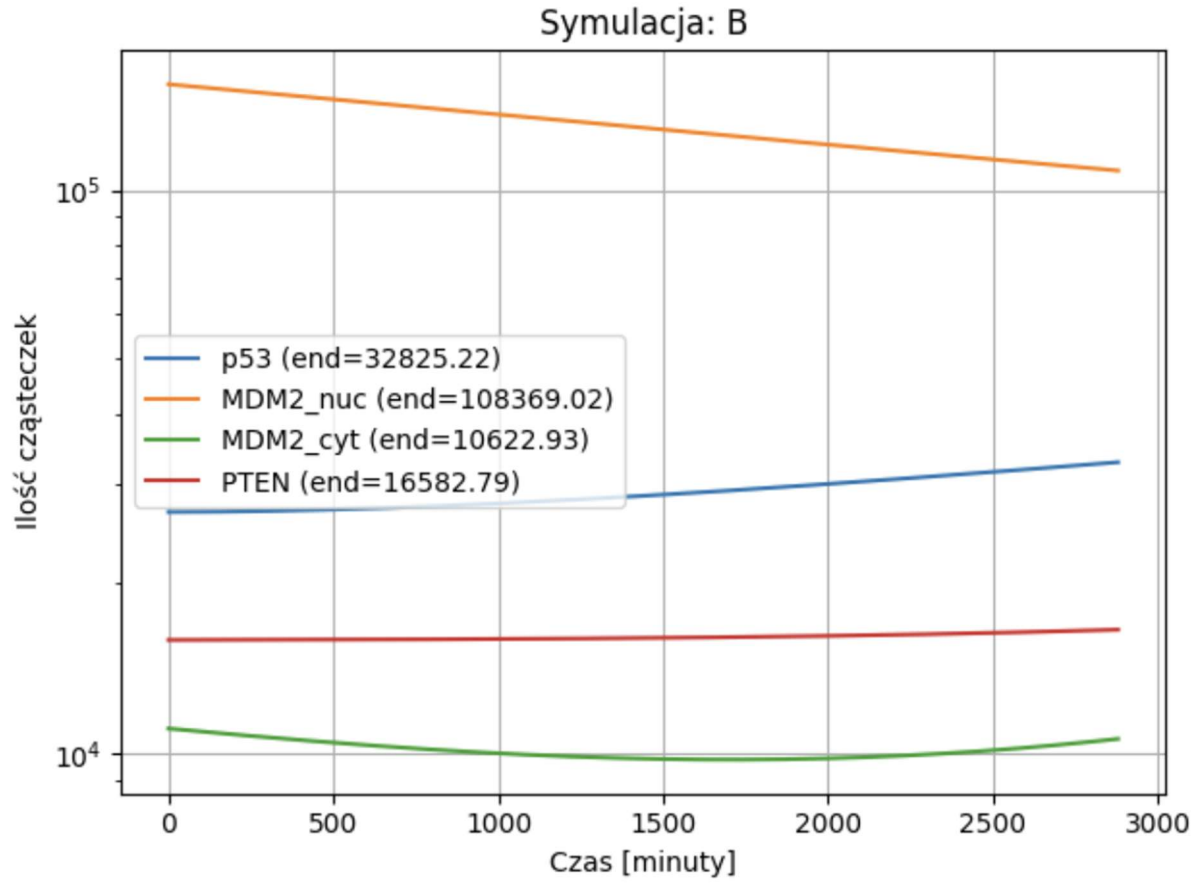



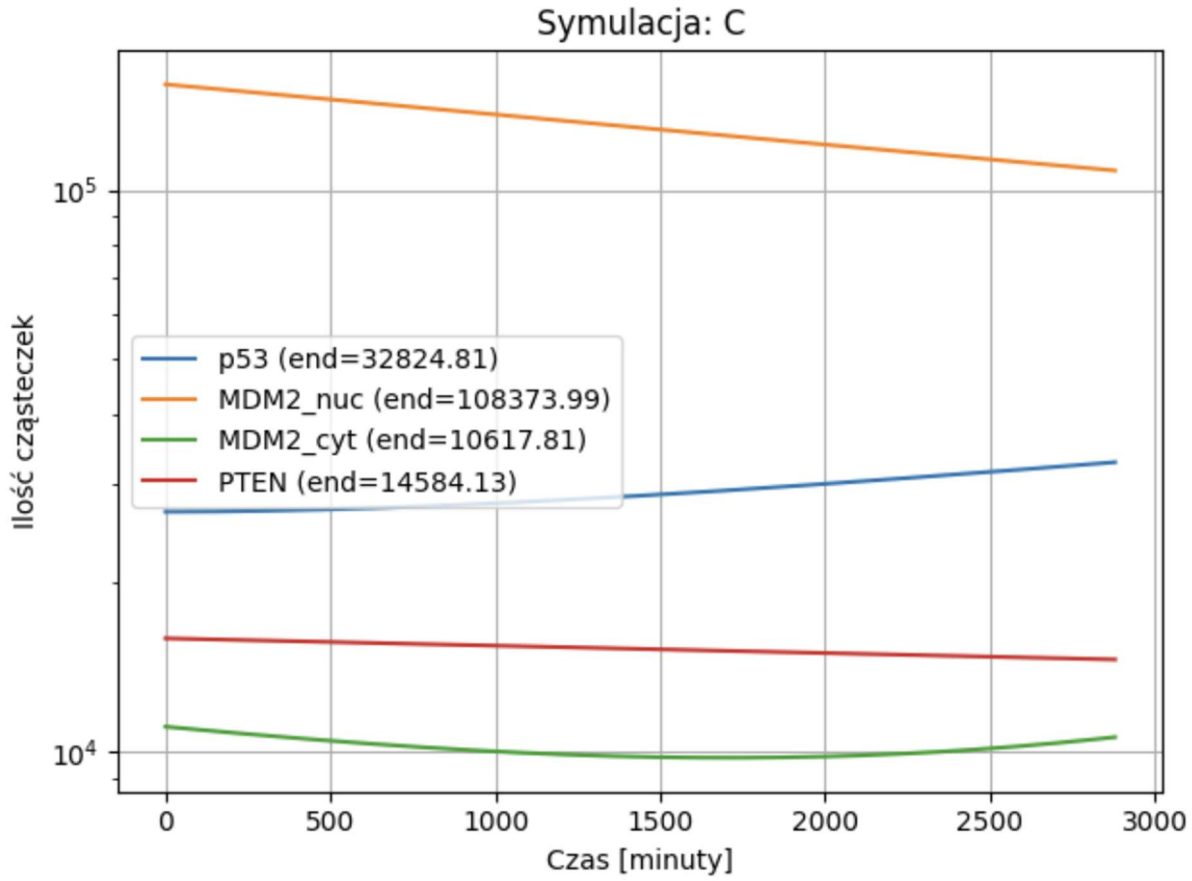

Symulacja: C
p53 (end=50862.80)
MDM2_nuc (end=116070.45)
MDM2_cyt (end=87934.63)
PTEN (end=0.49)
Ilość cząsteczek
Czas [minuty]
0
50000
100000
150000
200000
250000
300000
0
25000
50000
75000
100000
125000
150000
175000

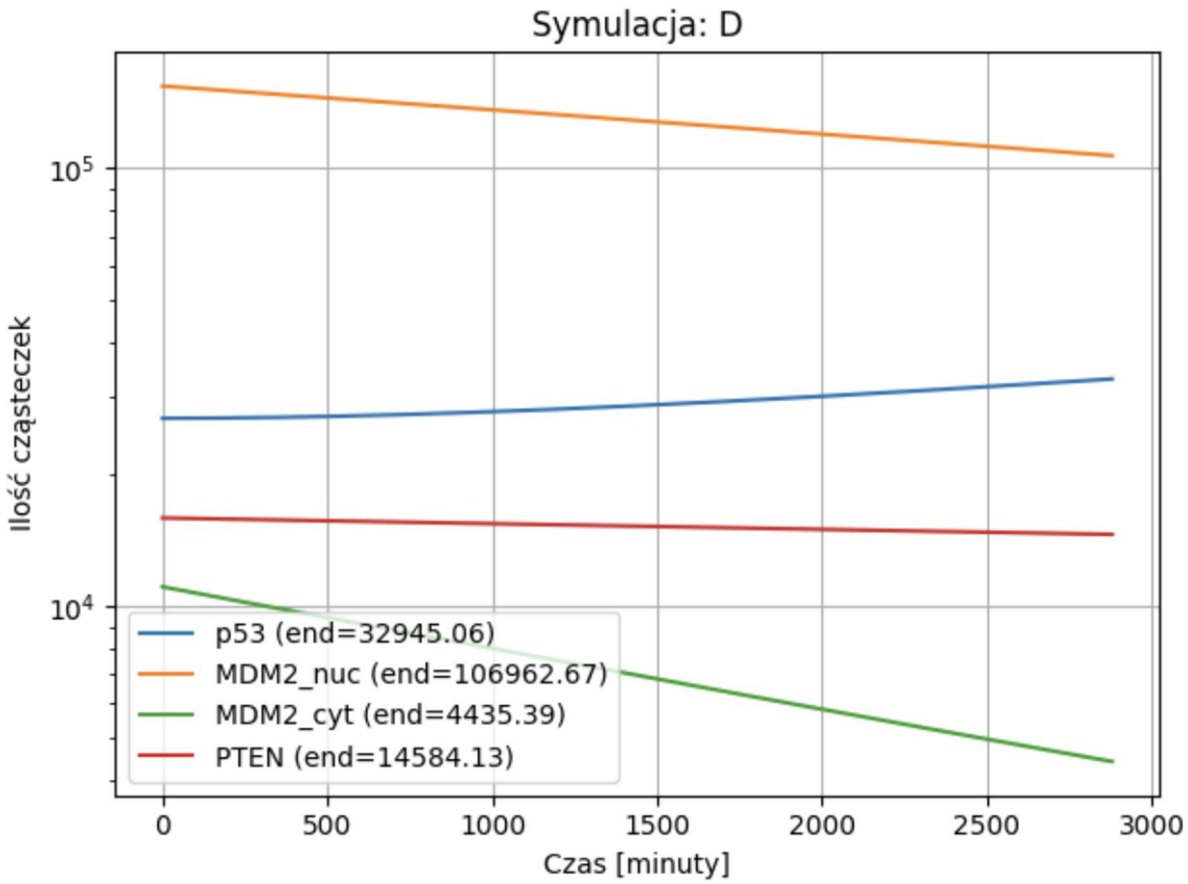

Symulacja: D
p53 (end=448584.10)
MDM2_nuc (end=37238.81)
MDM2_cyt (end=26600.50)
PTEN (end=0.49)
Ilość cząsteczek
Czas [minuty]
0
100000
200000
300000
400000
0
25000
50000
75000
100000
125000
150000
175000

**Tutaj symulacje po zmniejszeniu czasu do 2880 minut – zgodnie z poleceniem**




Symulacja: C
Ilość cząsteczek
$10^5$
$10^4$
p53 (end=32824.81)
MDM2_nuc (end=108373.99)
MDM2_cyt (end=10617.81)
PTEN (end=14584.13)
0
500
1000
1500
2000
2500
3000
Czas [minuty]


Symulacja: D
Ilość cząsteczek
$10^5$
$10^4$
p53 (end=32945.06)
MDM2_nuc (end=106962.67)
MDM2_cyt (end=4435.39)
PTEN (end=14584.13)
0
500
1000
1500
2000
2500
3000
Czas [minuty]

```python
import numpy as np
import matplotlib.pyplot as plt

# definicja wszystkich scenariuszy modelu w słowniku – dla każdego osobny wpis w słowniku, aby w razie potrzeby zmodyfikować parametry tylko dla wybranego scenariusza -> mało optymalne, ale było mi pomocne w trakcie pisania
BIO_SCENARIOS = {
    "A": {"params": {"p1": 8.8, "p2": 440, "p3": 100, "d1": 1.375e-14, "d2": 1.375e-4, "d3": 3e-5, "k1": 1.925e-4, "k2": 1e5, "k3": 1.5e5},
          "conditions": {"dna_damage": False, "siRNA": False, "pten_active": True},
          "initial": [26800, 154200, 11080, 15900]},
    "B": {"params": {"p1": 8.8, "p2": 440, "p3": 100, "d1": 1.375e-14, "d2": 1.375e-4, "d3": 3e-5, "k1": 1.925e-4, "k2": 1e5, "k3": 1.5e5},
          "conditions": {"dna_damage": True, "siRNA": False, "pten_active": True},
          "initial": [26800, 154200, 11080, 15900]},
    "C": {"params": {"p1": 8.8, "p2": 440, "p3": 100, "d1": 1.375e-14, "d2": 1.375e-4, "d3": 3e-5, "k1": 1.925e-4, "k2": 1e5, "k3": 1.5e5},
          "conditions": {"dna_damage": True, "siRNA": False, "pten_active": False},
          "initial": [26800, 154200, 11080, 15900]},
    "D": {"params": {"p1": 8.8, "p2": 440, "p3": 100, "d1": 1.375e-14, "d2": 1.375e-4, "d3": 3e-5, "k1": 1.925e-4, "k2": 1e5, "k3": 1.5e5},
          "conditions": {"dna_damage": True, "siRNA": True, "pten_active": False},
          "initial": [26800, 154200, 11080, 15900]}
}

# standardowa funkcja rk4
def simulate(system_eq, y_start, t_span, args):
    y_result = np.zeros((len(t_span), len(y_start)))
    y_result[0] = y_start
    for i in range(1, len(t_span)):
        dt = t_span[i] - t_span[i-1]
        y = y_result[i-1]
        k1 = system_eq(t_span[i-1], y, *args)
        k2 = system_eq(t_span[i-1] + dt/2, y + dt/2*k1, *args)
        k3 = system_eq(t_span[i-1] + dt/2, y + dt/2*k2, *args)
        k4 = system_eq(t_span[i-1] + dt, y + dt*k3, *args)
        y_result[i] = y + dt/6 * (k1 + 2*k2 + 2*k3 + k4)
    return y_result

# „regulacja” parametrów
def biological_dynamics(t, y, p, cond):
    p53, mdm2_n, mdm2_c, pten = y
    p_mod = p.copy()

# modyfikacje przy konkretnych warunkach, według wytycznych z polecenia
    if cond["siRNA"]:
        p_mod["p2"] *= 0.02
    if cond["pten_active"] == False:
        p_mod["p3"] = 0
    if not cond["dna_damage"]:
        p_mod["d2"] *= 0.1

# równania różniczkowe
    synthesis = p_mod["p1"]
```

```
    feedback = p_mod["d1"] * p53 * (mdm2_n ** 2)
    mdm2_prod = p_mod["p2"] * p53**4 / (p53**4 + p_mod["k2"]**4)
    pten_reg = p_mod["k3"]**2 / (p_mod["k3"]**2 + pten**2)

    dp53 = synthesis - feedback
    dmdm2_c = mdm2_prod - p_mod["k1"] * pten_reg * mdm2_c - p_mod["d2"] * mdm2_c
    dmdm2_n = p_mod["k1"] * pten_reg * mdm2_c - p_mod["d2"] * mdm2_n
    dpten = p_mod["p3"] * p53**4 / (p53**4 + p_mod["k2"]**4) - p_mod["d3"] * pten

    return np.array([dp53, dmdm2_n, dmdm2_c, dpten])

# tworzenie wykresów i zapisywanie
def generate_plot(time, data, title):
    labels = ["p53", "MDM2_nuc", "MDM2_cyt", "PTEN"]

    plt.figure()
    for idx, label in enumerate(labels):
        final_value = data[-1, idx]  # ostatnia wartość danej zmiennej, aby łatwo odczytać
parametry z wykresu
        label_with_value = f"{label} (end={final_value:.2f})"
        plt.plot(time, data[:, idx], label=label_with_value)

    plt.xlabel("Czas [minuty]")
    plt.ylabel("Ilość cząsteczek")
    plt.title(f"Symulacja: {title}")
    plt.grid(True)
    plt.legend()
    plt.yscale('log')
    plt.tight_layout()
    plt.savefig(f"symulacja_{title}.png")
    plt.close()

# główna funkcja
def run_all():
    time_minutes = np.arange(0, 2880, 0.5)  # 0.5 min = 30 sekund
    for scenario, config in BIO_SCENARIOS.items():
        y_initial = config["initial"]
        params = config["params"]
        flags = config["conditions"]
        result = simulate(biological_dynamics, y_initial, time_minutes, args=(params,
flags))
        generate_plot(time_minutes, result, scenario)

if __name__ == "__main__":
    run_all()
```