

# Kurs DevOps

## Lista 6

19 i 20 listopada 2025

Jako że zazwyczaj w dużych grupach znajduje się co najmniej jeden użytkownik Windowsa, to informuje się, iż zadania należy wykonywać z linii poleceń, o ile nie powiedziano inaczej w treści.

Używanie pluginów/rozszerzeń open-source do Ansible i Kubernetesa jest dozwolone, o ile w treści zadania nie powiedziano inaczej.

### Zadanie 1. (1.5 p.)

Zdefiniuj playbook Ansible, który w sytuacji, gdy nie będzie mógł połączyć się z agentem (kontenerem dockera), wykona akcję naprawczą w postaci uruchomienia brakującego kontenera. Czy w łatwy sposób można poprzedzić ponowne uruchomienie, sprawdzeniem, czy rzeczywiście dany kontener jest wyłączony?

Czy istnieje plugin do zarządzania inwentarzem Ansibla, który pozwala zintegrować go wprost z dockerem? Zademonstruj go.

### Zadanie 2. (2 p.)

Napisz swój własny moduł Ansible, który będzie wykonywał na zdalnym komputerze jedną z dwóch akcji (wybieranych z użyciem argumentu):

- Uruchomi pewną nietrywialną operację, której nie ma we wbudowanych modułach Ansible (np. fork bombę<sup>1</sup>).
- Rozwiąże problem n-hetmanów (dla n podanego jako argument).<sup>2</sup>

Pokaż, w jaki sposób dokumentuje się zawartość modułu.

### Zadanie 3.

Stwórz *playbook*, w którym połączysz się do zdalnej maszyny (np. kontenera dockera) i uruchomisz asynchroniczne nadpisywanie pliku zerami.<sup>3</sup> W trakcie nadpisywania pobierz równolegle, wybraną przez Ciebie stronę internetową. Poczekaj na zakończenie obu zadań przed przejściem do następnych tasków.

### Zadanie 4.

Czym są *roles* w Ansible? Zaprezentuj jak stworzyć przykładową *role* i jak ją następnie uruchomić.

### Zadanie 5. (1.5 p.)

Na podstawie poniższego skryptu do zgłaszania kodu źródłowego do oceny na zawodach programistycznych, stwórz szablon z użyciem Jinja2, a następnie użyj go<sup>4</sup> jako część *playbooka* konfigurującego stację zawodniczą.

- Skrypt powinien być przechowywany na węźle zarządzającym w postaci szablonu Jinja2.
- Podczas konfiguracji stacji zawodniczej szablon skryptu powinien być wypełniany z użyciem następujących danych: IP serwera HTTP, port serwera HTTP, imię i nazwisko użytkownika, identyfikator zawodnika, identyfikator stacji, klucz API unikalny dla danego zawodnika.
- Zapewnij – z co najmniej bardzo dużym prawdopodobieństwem – unikalność kluczy api (np. wygeneruj losowy ciąg znaków).

---

<sup>1</sup>Testując, możesz chcieć ograniczyć liczbę procesów, które można uruchomić w jednym kontenerze.

<sup>2</sup>Przy poszanowaniu prawa autorskiego, do rozwiązania problemu n-hetmanów, można wykorzystać kod innego studenta, bądź kod open-source.

<sup>3</sup>Na przykład z użyciem polecenia `dd`.

<sup>4</sup>W tym celu można na przykład użyć `ansible.builtin.template`.

- Klucze api powinny być przechowywane w pliku na hoście (więc jeśli są losowane w trakcie wykonania playbooka – co jest sugerowane w powyższym punkcie – to należy je zapisać do pliku).
- Po podstawieniu wartości w szablonie gotowy skrypt powinien zostać skopiowany do katalogu domowego na stacji zawodnika.
- Pokaż, że wygenerowany skrypt działa poprawnie.<sup>5</sup>

```
#!/bin/bash
temp="$(mktemp -d)"
cp "$1" "$temp"
cd "$temp"
staraNazwa="$(basename $1)"
rozszerzenie="${staraNazwa##*\.}"
nowaNazwa="jan-kowalski-id-123-stanowisko-14-$(date +%Y%m%d-%H%M%S).$rozszerzenie"
mv "$staraNazwa" "$nowaNazwa"
curl -H "X-Klucz:1234567abcd" -F "path=@$nowaNazwa" "http://127.0.0.1:8080/upload?path="
cd -
```

### Zadanie 6.

Pokaż, że playbook Ansible ma szansę<sup>6</sup> być maszyną Turinga, czyli że można zaimplementować nieskończoną pętlę (która wykonuje jakąś nietrywialną operację) lub pokaż, że można zaimplementować rekurencję.<sup>7</sup>

### Zadanie 7. ★

Bez tworzenia własnego modułu, zaimplementuj w Ansible playbooka, który odczyta z pliku opis skierowanego grafu nieważonego i wyznaczy długość najkrótszej ścieżki między dwoma zadanymi wierzchołkami.<sup>8</sup>

---

<sup>5</sup>Polecenie `curl` użyte w skrypcie jest dostosowane do współpracy z `miniserve -u`. Jeśli chcesz, możesz użyć innego serwera i odpowiednio zmodyfikować `curl`.

<sup>6</sup>Nie jesteśmy na JFIZO ani na TPJP, więc "ma szansę" jest wystarczające i nie będziemy się zajmowali formalnymi wymaganiami, które trzeba by było pokazać.

<sup>7</sup>Chcemy użyć wbudowanych mechanizmów Ansible. Odpowiedź, która wywołuje zewnętrzny program w języku Turing zupełnym, nie jest akceptowana.

<sup>8</sup>W tym zadaniu są zapewne co najmniej dwa podejścia, które mogą być warte rozważenia: a) użycie pętli i instrukcji imperatywnych dostarczanych przez Jinja2 b) wykorzystanie pętli/rekurencji z poprzedniego zadania.