Laboratorium 4 – Zmienne środowiskowe, argumenty linii komend, przetwarzanie plików

Języki skryptowe

Cele dydaktyczne

1. Zapoznanie ze zmiennymi środowiskowymi i czytaniem parametrów z linii komend w języku Python.
2. Zapoznanie z uruchomieniem procesów i komunikacji z nimi.
3. Zapoznanie z przetwarzaniem danych w formatach CSV oraz JSON.
4. Zapoznanie z operacjami na systemie plików.

Wprowadzenie

Zmienne środowiskowe

Istnieją różne sposoby na sterowaniem wykonaniem programów komputerowych. Zmienne środowiskowe są zmiennymi, których wartości są ustawiane poza programem, najczęściej przez funkcjonalności wbudowane w system operacyjny albo oprogramowanie zarządzające wykonywaniem usług. Zmienne środowiskowe składają się z par nazwa-wartość. Mogą przechowywać np. konfigurację aplikacji, co jest dobrą praktyką w tworzeniu aplikacji uruchamianych jako usługi.

Innym sposobem sterowania przebiegiem wykonania programu jest wykorzystanie argumentów linii komend. W języku Python dostępne są one na liście sys.argv. Pierwszy argument odpowiada nazwie skryptu, a kolejne reprezentują przekazane parametry. W kolejnych laboratoriach wykorzystane zostaną narzędzia wspierające tworzenie zaawansowanych CLI (ang. command-line interface).

Zadania

1. Napisz skrypt, który wyświetli na wyjście standardowe listę wszystkich zmiennych środowiskowych.
    a. Niech skrypt umożliwia uruchomienie go z dowolną liczbą parametrów linii komend. W takim przypadku, należy przefiltrować zmienne do wyświetlenia na wyjściu standardowym. Warunkiem wyświetlenia zmiennej i jej wartości jest istnienie parametru, którego wartość zawiera się w nazwie zmiennej.
    b. Zmienne powinny być wyświetlone w porządku alfabetycznym.
2. Napisz skrypt, który operuje na zmiennej środowiskowej PATH. Zmienna ta wykorzystywana jest w różnych systemach operacyjnych, m.in. Windows, Linux, Mac OS X. Zmienna ta zawiera katalogi, w których znajdują się pliki wykonywalne, które mogą być uruchamiane bez wpisywania pełnej ścieżki do pliku. Skrypt powinien umożliwić, z wykorzystaniem samodzielnie ustalonych parametrów linii komend, na realizację poniższych funkcjonalności :
    a. Wypisanie na wyjście standardowe wszystkich katalogów znajdujących się w zmiennej środowiskowej PATH, każdy w osobnej linii.
    b. Wypisanie na wyjście standardowe każdego katalogu znajdującego się w zmiennej środowiskowej PATH wraz z listą wszystkich plików wykonywalnych znajdujących się w tym katalogu.
3. Napisz program w ulubionym języku programowania (dowolnym, np. C, C++, Rust, Go, Java, Python, PHP, ...), który:
    a. czyta z wejścia standardowego ścieżkę do pliku tekstowego
    b. analizuje plik tekstowy pod kątem statystycznym, a następnie dla oblicza następujące informacje:
        i. ścieżka do pliku,
        ii. całkowita liczba znaków,
        iii. całkowita liczba słów,
        iv. liczba wierszy,
        v. znak występujący najczęściej,
        vi. słowo występujące najczęściej.
    c. wynik obliczeń wypisywany jest na wyjście standardowe powinien w formacie *.tsv, *.csv, lub *.json
    d. Następnie, napisz skrypt w języku Python, który:
        i. przyjmuje jako argument linii komend ścieżkę do katalogu w systemie plików,
        ii. z wykorzystaniem modułu subprocess uruchamia napisany powyżej

program do obliczeń, przesyłając na wejście standardowe ścieżki do kolejnych plików,

iii. przetwarza dane wyjściowe kolejnych wywołań programu, zapisując wynik jako listę słowników,

iv. wypisuje na wyjście standardowe w dowolnym formacie:

1. liczbę przeczytanych plików, sumaryczną liczbę znaków, sumaryczną liczbę słów, sumaryczną liczbę wierszy, znak występujący najczęściej, słowo występujące najczęściej.

4. Z wykorzystaniem programów i poleceń wybranego systemu operacyjnego (np. zip, tar, mv, cp, itd.) oraz modułu subprocess skonstruuj przedstawione poniżej skrypty. Zadbaj o to, by wspólne funkcjonalności dla obu skryptów zostały wyodrębnione do niezależnego modułu.

a. skrypt `backup.py` do tworzenia kopii zapasowych, który:

i. przyjmuje jako argument linii komend ścieżkę do katalogu w systemie plików,

ii. tworzy archiwum *.zip lub *.tar.gz zawierające wszystkie pliki z folderu przekazanego jako argument.

1. niech nazwa pliku ma postać `{timestamp}-{dirname}.{ext}`, gdzie:

a. timestamp - znacznik czasu zawierający kolejno kolejno rok, miesiąc, dzień, godzinę, minutę, sekundę utworzenia pliku,

b. dirname oznacza nazwę katalogu,

c. ext oznacza rozszerzenie pliku z archiwum.

iii. przenosi plik do katalogu o nazwie `.backups` w folderze użytkownika. Jeśli ten folder nie istnieje, program go tworzy.

iv. program pozwala na modyfikację lokalizacji katalogu `.backups` poprzez zmienną środowiskową `BACKUPS_DIR`. Przykładowo, wywołanie programu

```
BACKUPS_DIR=/home/user/alt_backups python backup.py data
```

spowoduje wykonanie kopii zapasowej plików z katalogu `data` oraz zapisanie jej w katalogu `/home/user/alt_backups`.

v. dodatkowo, w katalogu BACKUPS_DIR skrypt powinien utworzyć plik zawierający historię wykonanych kopii zapasowych.

1. Plik powinien mieć format *.csv lub *.json.

2. W pliku powinien zostać zapisany rekord w wybranym formacie

zawierający:
    a. datę wykonania kopii zapasowej,
    b. pełną lokalizację skopiowanego katalogu
    c. nazwę pliku z kopią zapasową.
3. W przypadku, gdy plik już istnieje, należy zmodyfikować plik, dopisując rekord do pliku.

b. skrypt `restore.py` do przywrócenia katalogu z kopii zapasowej;
    i. skrypt powinien przyjmować jako argument linii komend ścieżkę do katalogu w systemie plików,
        1. jeżeli ścieżka nie zostanie jawnie podana, należy przyjąć za ścieżkę obecny katalog roboczy.
    ii. program pozwala na modyfikację lokalizacji katalogu `.backups` poprzez zmienną środowiskową `BACKUPS_DIR`.
    iii. po uruchomieniu, skrypt powinien przeanalizować plik z historią wykonanych kopii zapasowych i wyświetlić użytkownikowi ich ponumerowaną listę w kolejności od najmłodszego do najstarszego.
    iv. użytkownik powinien wskazać wybraną kopię zapasową do przywrócenia, podając jej numer na wejście standardowe.
    v. po wybraniu kopii, skrypt powinien usunąć zawartość katalogu, a następnie rozpakować archiwum zawierające wybraną kopię zapasową do podanej ścieżki.