Programowanie I R

Zadania – seria 14.

Programowanie współbieżne i asynchroniczne.

Zadanie 1. `callspeed` – Czas współbieżnego i równoległego wywołania funkcji.

Napisz funkcję `fcall`, która przyjmuje jako argumenty wywołania liczbę naturalną n, pewną funkcję oraz ewentualne argumenty wywołania tej funkcji. Funkcja `fcall` powinna n-krotnie wywołać funkcję przekazaną jej jako drugi argument z ewentualnymi jej argumentami i zwrócić łączny czas trwania wszystkich n wykonań tej funkcji.

Napisz także funkcje `fcall_Thread` i `fcall_Process`, działające podobnie, jak funkcja `fcall`, funkcja `fcall_Thread` powinna jednak realizować każde z n wykonań funkcji będącej jej argumentem w osobnym wątku, zaś funkcja `fcall_Process` – w osobnym procesie.

Przygotuj ponadto kilka funkcji realizujących dowolne czasochłonne zadania (np. tworzenie dużej listy liczb pseudolosowych i sortowanie jej za pomocą metody `sort`, tworzenie takiej listy i sortowanie jej metodą bąbelkową, tworzenie macierzy liczb pseudolosowych i jej diagonalizacja z wykorzystaniem pakietu NumPy). Funkcje powinny być napisane tak, by ich wykonanie zajmowało co najmniej kilka sekund.

Korzystając ze wszystkich tych funkcji, napisz program `callspeed`, przyjmujący jako argument wywołania liczbę naturalną k. Program powinien wykonać każdą z napisanych przez Ciebie czasochłonnych funkcji na trzy sposoby, używając do tego funkcji `fcall`, `fcall_Thread` i `fcall_Process` z liczbą wykonań równą k, i wypisać na ekranie czas trwania tych wykonań w każdym przypadku.

Zadanie 2. `linkscraper` – Wydobywanie hiperłączy ze stron internetowych.

Napisz program `linkscraper`, przyjmujący dwa argumenty wywołania: łańcuch tekstowy reprezentujący adres URL strony internetowej oraz liczbę naturalną. Program powinien przeanalizować kod HTML strony o podanym adresie i odnaleźć w nim wszystkie hiperłącza (reprezentowane przez znaczniki `a` z atrybutem `href`) do stron internetowych (adres zaczyna się od `http://` lub `https://`), a następnie dla każdego z nich wypisać adres strony, na którą wskazuje (czyli wartość atrybutu `href`).

Gdy liczba naturalna przekazana programowi jako drugi argument wywołania jest równa 1, po wykonaniu tego zadania praca programu powinna się zakończyć. Jeśli liczba ta jest równa 2, program powinien wykonać to samo zadanie dla wszystkich stron, do których odnośniki znalazł na przeanalizowanej stronie; gdy jest ona równa 3, program powinien wykonać to samo zadanie również dla stron, do których odnośniki znalazł na stronach linkowanych na pierwotnej stronie etc.

Program powinien wykorzystywać współbieżność: każda ze stron powinna zostać pobrana i przeanalizowana w osobnym wątku.

Wskazówka. Kod HTML strony WWW o zadanym adresie URL możesz uzyskać, posługując się na przykład modułem `urllib.request`. Hiperłącza występujące w kodzie HTML najłatwiej odszukać posługując się wyrażeniami regularnymi lub parserem HTML (np. *Beautiful Soup*).

Zadanie 3. `imgscraper` – Asynchroniczne pobieranie obrazów ze strony WWW.

Napisz program `imgscraper`, przyjmujący dwa argumenty wywołania: łańcuch tekstowy reprezentujący adres URL strony internetowej oraz łańcuch tekstowy reprezentujący ścieżkę do folderu. Program powinien przeanalizować kod HTML strony o podanym adresie i odnaleźć w nim wszystkie obrazy (reprezentowane przez znaczniki `img`, adres pliku z obrazem jest wartością atrybutu `src`), a następnie pobrać każdy z nich i zapisać we wskazanym folderze.

Program powinien wykonywać operacje wejścia/wyjścia asynchronicznie; dotyczy to zarówno komunikacji z analizowaną stroną WWW oraz pobierania i zapisywania obrazów.

Wskazówka. Programowanie asynchroniczne jest możliwe dzięki modułowi `asyncio`. Wraz z nim powinieneś użyć modułu `aiohttp` do asynchronicznej komunikacji ze stroną internetową oraz modułu `aiofiles` do asynchronicznej obsługi systemu plików. Obrazy występujące w kodzie HTML najłatwiej odszukać posługując się parserem HTML (np. *Beautiful Soup*).

Opracowanie: Bartłomiej Zglinicki.