Отримані дані

Дані, які отримуються з інтернету, а саме з Udemy API, представлені через бібліотеку request, а згодом оброблені за допомогою бібліотеки json, і перетворюються в клас словник, де ключами є такі значення: кількість курсів, які підійшли під задані параметри, посилання на наступну “пачку” курсів, тобто є максимальна кількість курсів, які можуть бути отримані за один запит, також посилання на минулу “пачку” і власне результат, що є списком, який складається з словників, де описані курси – це є відповідь на запит get\_courses\_list. Проте є ще один запит - get\_curriculum\_items – це дані про оприділений курс. Тут все побудована майже таким самим чином, як в get\_courses\_list, проте в results представлені не словники даних про курси, а словники даних про розділи та пдрозділі оприділеного курсу.

Дальше йде опрацювання всіх даних – для цього є написані такі функції : check\_category та check з модуля пакунків checker.

Check\_category – це функція, яка перевіряє, чи користувач вказав існуючу категорію курсів чи ні, для цього вона перевіряє чи введена категорія є в списку всіх категорій чи ні, якщо є то вона повертає назву категорії, якщо ні то вона перевіряє чи є в списку категорія подібна до введеної, якщо подіність досить велика, то вона повертає категорію зі списку, якщо ні то вона виводить всі можливі категорії і піднімає помилку, в подальшому ця помилка ловиться і користувач ще раз вводить назву категорії.

Check – це функція, яка приймає список з словниками про курси, дальше програма запитує користувача необхідні дані і згодом просить в спецільний файл формата \*.txt записати всі глави та теми, кожна з нового рядка, після чого користувач ввадить done і функція запускає перевірку, через що цей методі працює довше, проте більша ймовірність що виведені курси будуть відповідати введеним даних користувача.

Write – записує в файл отримані посилання на курси.

Read – зчитує порядково інформацію з файла.

Це були функція, для опрацювання даних, якщо користувач вибере методи написані мною, проте є ше варіант, якщо він вибере методи самого Udemy API. Для цього реалізовано функцію make\_url.

Make\_url – питає користувача про подробиці курсу, після чого робить екземпляр класу URL. Тобто користувач має ввести всі необхідна параметри такі, як наприклад ціна, якщо користувач вводить free добаляється до посилання price=price-Free, якщо ж платне то price=price-Paid, якщо ж нічого, то нічого не додається.

Абстрактні типи даних

Вся суть абстрактного типу даних полягає в тому що користувачі використовують його через інтерфейси, не дивлячись на реалізацію, оскільки остання з часом може змінитися. Тобто розробнику програмного забезпечення не обовязково знати як реалізований ADT, йому потрібно знати тільки, що приймає і що видає ADT. До того ж надзвичайно легко виправлятиб дебажити абстрактний тип даних.

Написаний мною ADT називається URL і знаходиться в папці модулів URL. Даний клас відповідає за побудову посилання, за яким в подальшому буде братися інформація про курси. На початку url=”[https://www.udemy.com/api-2.0/courses/?page size=100](https://www.udemy.com/api-2.0/courses/?page%20size=100)”

Current\_url=<https://www.udemy.com/api-2.0/courses/id/public-curriculum-items/?page_size=100>

Дальше є декілька методів, які добавляють свої символи до загальної url

add\_category() – додає категорію до url, наприклад якщо ввести programming в параметрах до url буде додано &category = programming

add\_subcategory()– додає підкатегорію до url, наприклад якщо ввести programming в параметрах до url буде додано &subcategory = programming

price() - додає ціну до url, наприклад якщо ввести free, до url буде додано &price=price-free

is\_affiliate\_agreed() – додає чи узгоджено партнера до url, якщо так то до url буде додано &is\_affiliate\_agreed=True

is\_fixed\_priced\_deals\_agreed() - додає до url чи ціна є фіксованою чи ні, якщо так то дадає до url &is\_fix\_priced\_deals\_agreed=True

language() – додає до url мову, якщо мова наприклад українська, то до поточної url буде додано &language=ukr

has\_simple\_quiz() - додає до url чи курс має прості тести чи немає, тобто якщо він має до url буде додано &has\_simple\_quiz=True

has\_closed\_caption() - додає до url чи курс має закритий заголовок чи немає, тобто якщо він має до url буде додано &has\_closed\_caption=True

has\_coding\_exercises() - додає до url чи курс має вправи чи немає, тобто якщо він має до url буде додано &has\_coding\_exercises=True

instructional\_level() – додає до url рівень який мають мати повернені курси, наприклад якщо ви ввели beginner, то добавить &instructional\_level=Beginner

ordering() – показує статистику курсу, тобто яку статистику мають мати повернені курси, якщо ввести newest, буде дабавлено &ordering=newest

url() - повертає url, яку буде використано для пошуку courses list

curriculum\_url() - повертає url, яку буде використано для пошуку curriculum list course