Progr@muj w zespole - podział materiału

zmiany do 3 etapu - tylko lekcje w szkole

Ogólne założenia:

- 7 spotkań w szkole, w tym 5 poświęconych programowaniu
- 4 samodzielne prace uczniów w domu między spotkaniami poświęconymi programowaniu w szkole

Dla chętnych na koniec zapakowanie projektu do http://www.pyinstaller.org/

Podział materiału na lekcjach:

1. Lekcja w szkole - podział, tematy itp... wstęp
2. Lekcja w szkole - ustawienia wstępne środowiska
    - Praca samodzielna - maszyna wirtualna i podstawy aplikacji
3. Lekcja w szkole - podstawowe typy danych i konstrukcje programistyczne w Python
    - Praca samodzielna - obsługa głównych elementów biblioteki PySimpleGUI
4. Lekcja w szkole - requests i API - słowniki i JSON
    - Praca samodzielna - samodzielne testy dostępu do API
5. Lekcja w szkole - różne interfejsy aplikacji, `Commit/Push` do repozytorium
    - Praca samodzielna - definiowanie funkcji w Python i dalsze przygotowywanie dokumentacji
6. Lekcja w szkole - praca z kluczami i wartościami słowników
    - Praca samodzielna - końcowe tworzenie dokumentacji
7. Lekcja w szkole - podsumowanie projektów, wybór najlepszego projektu, post-testy itp.

Pamiętamy, aby po każdej lekcji w szkole lub pracy samodzielnej uczestnicy zaktualizowali swoje repozytoria poprzez `Commit/Push`

Podział filmów - z tego wynikną lekcje....

####

1. Konfiguracja IDE PyCharm, tworzenie konta w GitHub.com a01
2. Przygotowujemy środowisko `venv` dla lokalnego projektu a02
3. Plik `requirements.txt` - zewnętrzne moduły, własne pliki `py` w projekcie a03
4. Minimalny program z wykorzystaniem PySimpleGUI a04
5. Skąd będziemy czerpać grafiki? Pixabay, Freepik i Flaticon. (b01)
6. Uruchamiamy edytor tekstów - format strony. b02
7. Uruchamiamy edytor grafiki - otwieramy przykładowy plik graficzny. b03
8. Uruchamiamy edytor HTML - podstawowa strona z szablonu. b04
9. Edytor HTML: Bootstrap - https://getbootstrap.com/ b05

1. VirtualBox w Windows i jak importować maszynę `OVA` - aby pracować niezależnie od szkoły b06
2. PySimpleGui - dokumentacja, przykłady użycia a05
3. Edytor teskstów: nagłówek i stopka b07
4. Edytor teskstów: style i spis treści b08
5. Edytor grafiki: zmiana rozmiaru i zapis XCF b09
6. Edytor HTML: różne znaczniki `meta` b10

Lekcja w szkole (2):

1. Podstawowe typy danych w Python, zmienne a06
2. Typy zaawansowane: listy, słowniki a07
3. Importowanie z zewnętrznych modułów a08
4. Pętla `for` i listy a09
5. Instrukcja warunkowa `if ... else ...` a10
6. Edytor tekstów: listy numerowane i nienumerowane b11
7. Edytor grafiki: warstwy i dodanie elementu b12
8. Edytor HTML: containers b13

Praca samodzielna (2):

1. Wyświetlamy informację. a11
2. Rozpakowywanie tupli - pythonizm. a12
3. Pętla `while True` - sterowanie programem PySimpleGUI. a13
4. Dodajemy elementy przycisków. a14
5. Dodajemy wyświetlanie obrazków. a15
6. Sposoby wprowadzania danych. a16
7. Poznajemy sterowanie. a17
8. PySimpleGui - tworzymy prosty program okienkowy - 1 (layout, listy) a18
9. Wyświetlanie większej ilości danych (output) a19
10. Edytor teskstów: zrzut zawartości okna aplikacji i dodanie do tekstu b14
11. Edytor grafiki: warstwy i dodanie tekstu. b15
12. Edytor HTML: różne elementy na stronie (display, images, listy). b16

Lekcja w szkole (3):

1. Poznajemy `Python Console` w PyCharm + Wykorzystujemy `requirements.txt` i instalujemy niezbędne elementy: `requests`
   a20
2. Wykonujemy `request.get()` - z serwisu https://fastapi.jurkiewicz.tech/ pokazujemy odczytane dane a21
3. `JSON` i słowniki w Python a22
4. Użycie pętli `for` dla pokazania elementów słownika z serwisu https://fastapi.jurkiewicz.tech/ a23
5. Listy jako elementy słowników a24
6. Słowniki jako elementy słowników a25
7. Edytor grafiki: eksport obrazu jako PNG b17
8. Edytor tekstów: eksport dokumentu do formatu PDF b18
9. Edytor HTML: różne elementy na stronie (navbar, listy, image). b19

Praca samodzielna (3):

1. Sprawdzamy dokumentację dla przykładowych API: a26/ a27/ a28
    1. https://aviationstack.com/documentation
    2. https://numverify.com/documentation
    3. https://wttr.in/:help
2. Generowanie API_KEY dla wybranego projektu (Aviationstack) - https://aviationstack.com/signup/free a29
3. Generowanie API_KEY dla wybranego projektu (Numerify) - https://numverify.com/documentation a30
4. Poznajemy kody odpowiedzi API: poprawnych i błędnych a31 / a32
5. tworzymy własne repozytorium, pamiętamy o `.gitignore`, `README.md` oraz licencji a33

Lekcja w szkole (4):

1. Definiowanie funkcji w Python. a34
2. Funkcje i zasięg zmiennych w Python. a35
3. Testujemy dostęp do danych API (aviationstack) a36
4. Testujemy dostęp do danych API (numverify) a37
5. Testujemy dostęp do danych API (wttr.in) a38
6. Replikacja projektu z GitHub do PyCharm (open via VCS) i dodanie lokalnego venv (Add interpreter) i w requirements.txt `Install all packages` a39
7. Dodanie do repozytorium pracy z aplikacją PySimpleGUI i `Commit/Push` a40

Praca samodzielna (4):

1. Sprawdzenie działania — skrypt odczytujący i prezentujący wybrane dane a41
2. Wysłanie projektu do serwisu GitHub a42

Lekcja w szkole (5):

1. Aktualizowanie wartości dla kluczy słowników a43
2. Tworzenie słowników i dodawania do nich elementów a44

Praca samodzielna (5):

1. Tworzenie pełnej aplikacji
2. Tworzenie docelowego interfejsu aplikacji
3. Przygotowanie końcowe dokumentacji i strony w HTML