# DOKUMENTACJA PROJEKTU W RAMACH PRZEDMIOTU PROGRAMOWANIE W JĘZYKU PYTHON.

**Autor:**

Filip Maciborski

**Tytuł projektu:**

Vocabulary tester

**Tematy uwzględnione w projekcie:**

1. Git.
2. Czytelny kod w Pythonie – PEP8.
3. Zaawansowane struktury danych – moduł Pandas.
4. Obsługa dat i czasu – moduł DateTime.
5. Programowanie funkcyjne.
6. Programowanie obiektowe.

## Git i GitHub

Projekt od pewnego momentu przechowywany był na GitHubie i także za jego pośrednictwem został przekazany wykładowcy. W trakcie pracy nad projektem, choć dopiero od pewnego momentu stosowany był Git.

Git został zainstalowany oraz odpowiednio skonfigurowany. Został także połączony poprzez klucz SSH z GitHubem.

Obraz zawierający tekst, elektronika, zrzut ekranu, oprogramowanie

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.Historia commitów


## Czytelny kod w Pythonie – PEP8.

Nad projektem pracowałem z wykorzystaniem Visual Studio Code. Jako dodatek zainstalowałem *Flake8* podkreślający wszystkie możliwe błędy jakie występują w kodzie niezgodnym z *PEP8* np.:

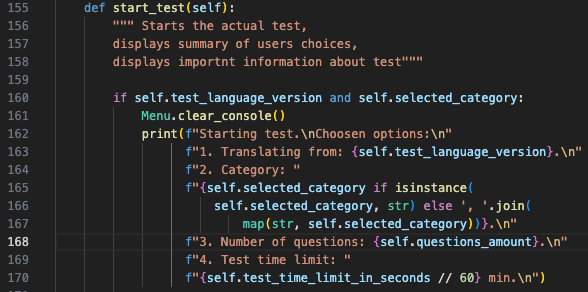
Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

Zastosowałem się do wszystkich wskazań dodatku i dostosowałem kod do *PEP8*.

**Przykłady** (na podstawie klasy *NewTest*)**:**

* + Dzielenie długich stringów na linie i stosowanie odpowiednich wcięć(klasa *NewTest*):



* + ‘Zawijanie’ długich wywołań funkcji z zachowaniem wcięć:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* + Komentarze opisujące zadania/działanie metod jako multiline doc-string:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* + Prawidłowe uporządkowanie argumentów funkcji w formie listy (klasa *NewTest*):

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* + Uporządkowane importy (klasa *NewTest*):

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* + Prawidłowe nazewnictwo klas i metod:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

## Zaawansowane struktury danych – moduł Pandas.

W projekcie został wykorzystany moduł *Pandas*. W największym zakresie wykorzystana została *DataFrame* w oparciu, o który program funkcjonuje:

* + Praca z plikiem Excel i arkuszami pliku w których została utworzona baza danych programu (klasa *AppController*):

Pobranie danych z pliku:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

Zapis danych do pliku:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* + Wykorzystanie *DataFrame* do zarządzania przepływem danych, podczas tworzenia pytań i zbierania odpowiedzi użytkownika, dodawania nowych danych do bazy danych.

Łączenie danych:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

Tworzenie nowej kategorii słów:  
Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

Operacje na kolumnach w celu pozyskania wyników testu (klasa *NewTest*):

- **map()** – zmiana wartości bazując w serii na podstawie podanej innej wartości

- **apply()** – stosowanie funkcji do wierszy lub kolumn

- **astype()** - zmiana typu danych w series

- **mean()** – średnia procentowa kolumny

- **sum()** – obliczenie sumy kolumny

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

## Obsługa dat i czasu – moduł DateTime.

Moduł *DateTime* został w programie wykorzystany do pobrania daty i godziny przeprowadzenia testu, a także zmierzenia czasu jego trwania. W oparciu o limit czasu ustawiany przez użytkownika w minutach, obliczany jest limit czasu na sekundy. Stworzony dekorator mierzy czas trwania testu. Na podstawie porównania wartości określane jest to czy użytkownik zmieścił się w limicie czasowym.

**Przykłady** (klasa NewTest)**:**

* + Pobranie daty i czasu podczas tworzenia instancji klasy *NewTest* – wybranie   
    z menu głównego opcji -> *‘Start New Test’*

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* + Ustalenie limitu czasu dla testu – bez wykorzystania *DateTime,* ale ważne   
    w późniejszym użyciu (klasa *TimeManager*)

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* + Pomiar czasu trwania testu (wrapper w *TimeManager*):

**Obraz zawierający tekst, zrzut ekranu, Czcionka

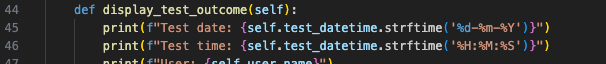
Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.**

* + Pomiar czasu uruchomiony dla funkcji, która wyświetla pytania i pobiera odpowiedzi użytkownika, czyli w momencie, gdy użytkownik przechodzi do rozwiązywania testu (klasa *NewTest*, metoda *def submit\_answer()* wywołana   
    w metodzie def *start\_test()*):

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.**

* + Wyświetlanie poszczególnych elementów *DateTime*, osobno daty   
    i czasu w odpowiednim formacie (klasa *ResultManager* metoda   
    *def display\_test\_outcome()*):

****

## Programowanie funkcyjne.

Program wykorzystuje paradygmat programowania funkcyjnego. Jest to poniekąd efekt tego, że zastosowałem podejście obiektowe.

* Funkcja wyższego rzędu – funkcja przyjmująca inną funkcję jako argument (wrapper w klasie *TimeManager*):

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.**

* Funkcja wyższego rzędu, przyjmuje tekst i kolor jako argument, a następnie zwraca zmodyfikowany tekst – przetwarzanie danych w stylu funkcyjnym (klasa *TextFormatter*).

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* Funkcja czysta, nie modyfikuje stanu obiektu, wynik funkcji zależy wyłącznie od przekazanych argumentów, zwraca ten sam wynik dla tych samych danych wejściowych (klasa *TextFormatter*):

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* Funkcja czysta, jedynym efektem jest wykonanie polecenia systemowego,   
  nie zależy od stanu obiektu i go nie modyfikuje (klasa *Menu*):

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

## Programowanie obiektowe.

Program wykorzystuje kilka cech programowania obiektowego. M.in.:

* Interfejsy i ich implementacja. Wykorzystane zostały dwa interfejsy (klasy abstrakcyjne), które wymuszą implementacje konkretnych metod podczas tworzenia klasy implementującej interfejs:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.



* Enkapsulacja. Zarówno niektóre metody oraz pola klas ukrywane są przed użytkownikiem i dostęp do nich jest ograniczony, gdyż z punktu widzenia użytkownika szczegóły implementacji nie są istotne lub ważne jest by użytkownik lub inne klasy nie mogły ich swobodnie modyfikować (klasa *NewTest*):

Obraz zawierający tekst, zrzut ekranu, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

Użytkownik nie ma w ogóle możliwości modyfikacji np. *\_\_self.point\_score* lub *self.\_\_percentage\_score*. Pola te modyfikowane są przez odpowiednie metody w klasie.  
  
Użytkownik ma możliwość modyfikowania pól *self.\_\_user\_name* oraz *self.\_\_test\_language* ale odbywa się to za pośrednictwem odpowiednich metod, które maja dostęp do tych pól. Same metody są również prywatne.

* W tym przypadku niektóre klasy korzystają z metod wewnętrznie i nie ma potrzeby by były dostępne spoza klasy. Metoda *def \_\_initiate\_language\_menu()* wywoływana jest w konstruktorze i służy do utworzenia menu wyboru języka. Żadna inna klasa nie musi z tej metody korzystać.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

* Kompozycja. Najlepszym przykładem jest klasa *NewTest*, która korzysta z kilku innych klas w celu obsługi najważniejszych elementów testu:

Obraz zawierający zrzut ekranu, tekst, Czcionka

Zawartość wygenerowana przez sztuczną inteligencję może być niepoprawna.

Klasa ta w konstruktorze inicjuje kilka obiektów które realizują:

1. **TextFormatter** – kolorowanie i normalizacja tekstu odpowiedzi, tak aby odpowiedzi mogły być udzielane w języku polskim bez ‘polskich znaków’
2. **Questionmanager** – odpowiada za określenie ilości pytań i stworzenie list pytań   
   i odpowiedzi.
3. **TimeManager** – odpowiada za ustalanie limitu czasu testu, mierzenie czasu przebiegu testu, odliczanie do rozpoczęcia testu i utrzymanie uśpienia ekranu np. podczas wyświetlania komunikatów o błędach
4. **FileManager** – odpowiada za zapis danych do pliku
5. **UserManager** – odpowiada za interakcję z użytkownikiem, ustawienie imienia, walidację wprowadzanych przez niego wartości w terminalu.