



Wydział Geodezji i Kartografii

POLITECHNIKA WARSZAWSKA

TRANSFORMACJE MIĘDZY RÓŻNYMI UKŁADAMI GEODEZYJNYMI

INFORMATYKA GEODEZYJNA
SEM. IV, ĆWICZENIA, ROK AKAD. 2022-2023

ADRIAN MAKSYMIOUK
GRUPA II, NUMERY INDEKSU: 319345
DAWID JUNDO
GRUPA II, NUMERY INDEKSU: 319328
01169882@pw.edu.pl lub 01169863@pw.edu.pl

WYDZIAŁ GEODEZJI I KARTOGRAFII
POLITECHNIKA WARSZAWSKA
Zakład Geodezji Wyższej i Astronomii
Warszawa, 29 kwietnia 2023

Spis treści

1	Cel ćwiczenia	2
2	Wykorzystane narzędzia i materiały potrzebne do replikacji ćwiczenia	2
2.1	Wybrany język programowania i interpreter Spyder	2
2.2	System operacyjny	2
2.3	Potrzebne biblioteki i pliki	2
3	Przebieg ćwiczenia	3
3.1	Stworzenie klasy Transformacja	3
3.2	Algorytm hirvonena	3
3.3	flh2XYZ	3
3.4	fl2PL1992	3
3.5	fl2PL2000	3
3.6	xyz2neu	3
3.7	Wczytywanie i zapisywanie pliku	4
3.8	Testy dla funkcji	4
3.9	Kalkulatory transformacji i zapis ich wyników do Kalkulatora	4
3.10	Dodanie możliwości wczytania pliku w argparse	4
3.11	Dodanie graficznego interfejsu GUI	4
4	Podsumowanie	5
4.1	Rezultaty	5
4.2	Umiejętności nabyte	5
4.3	Spostrzerzenia, problemy i ich rozwiązania:	5

1 Cel ćwiczenia

Celem ćwiczenia jest stworzenie skryptu w języku programowania Python w postaci klasy zawierającej metody, które implementują poszczególne transformacje z układu współrzędnych kartezjańskiego (x , y , z) lub układu geodezyjnego (ϕ , λ , H). W tym celu należy zastosować odpowiednie algorytmy lub metody przekształcenia współrzędnych do innych układów.

- `hirvonen(xyz2flh)`
- `flh2xyz`
- `flh2PL92`
- `flh2PL`
- `xyz2neu`

2 Wykorzystane narzędzia i materiały potrzebne do replikacji ćwiczenia

2.1 Wybrany język programowania i interpreter Spyder

- Python - język programowania, w którym napisany jest skrypt ćwiczenia.
- Spyder - jest to środowisko programistyczne dla języka Python, które zawiera edytor kodu, interpreter, konsolę i wiele innych funkcjonalności.
- Najlepiej pobrać Spydera za pośrednictwem anacondy, która ma domyślnie zainstalowane środowisko programistyczne Spyder www.anaconda.com/download (Wang i Oliphant, 2012)

2.2 System operacyjny

Ten skrypt został napisany w systemie operacyjnym Microsoft (Windows 10 oraz Windows 11).

2.3 Potrzebne biblioteki i pliki

Do wykonania ćwiczenia należy użyć następujących bibliotek:

1. Numpy - to biblioteka w języku Python, która służy do obliczeń numerycznych i analizy danych. Numpy dostarcza wiele narzędzi do pracy z wielowymiarowymi tablicami danych oraz narzędzi do wykonywania operacji matematycznych i statystycznych na tych tablicach. Numpy nie jest wbudowany w Pythona, ale jest dostarczony z Anacondą, co oznacza łatwość dostępu.
2. Argparse - to biblioteka w języku Python, która służy do parsowania argumentów linii poleceń. Argparse jest częścią standardowej biblioteki Pythona co oznacza że jest wbudowany w standardową instalację Anacondy.
3. Pytest to biblioteka w języku Python, która służy do testowania kodu źródłowego. Umożliwia łatwe i elastyczne pisanie testów. Pytest nie jest wbudowany w standardową instalację Pythona ani w dystrybucji pakietów Anaconda, ale można ją zainstalować za pomocą menadżera pakietów pip.
4. Os to biblioteka standardowa w języku Python, która zapewnia interfejs do operacji na systemie operacyjnym (np. dostęp do plików, zarządzanie procesami, zmiana katalogu roboczego itp.).
5. Tkinter to biblioteka graficzna dla języka programowania Python. Biblioteka ta umożliwia tworzenie interfejsów graficznych użytkownika (GUI) dla programów Python. Tkinter jest dostępny w standardowej bibliotece Pythona i jest łatwo dostępny na większości platform.

Należy również pobrać plik tekstowy o nazwie "wsp_inp.txt", który znajduje się na zdalnym repozytorium GitHub pod linkiem: https://github.com/Sawoboh/Informatyka_Projek_1.git da on możliwość wczytania i wykonania transformacji z zawierających danych. Następnie zapisze te dane do pliku wynikowego.

3 Przebieg ćwiczenia

3.1 Stworzenie klasy Transformacja

Stworzono klasę Transformacje oraz `__init__` w której podano parametry elipsoidy (a , e^2) dla jakich można wykonać obliczenia. Trzeba pamiętać że za każdym razem gdy się będzie odwoływać do parametrów elipsoidy należy poprzedzić to `self`. Następnie dodano przypadek w którym podane zostanie zła nazwa elipsoidy. Wtedy wyskoczy błąd.

3.2 Algorytm hirvonena

Algorytm hirvonena przelicza współrzędne kartezjańskie (x y z) na współrzędne geodezyjne (ϕ , λ , H). Algorytm był używany na zajęciach z Geodezji wyższej. Wtedy poznano idee tego rozwiązania. Natomiast do przypomnienia skorzystano z strony www.asgeupos.pl - pdf z geodezyjnymi transformacjami. (Kadaś, 2002) W funkcji implementującej to rozwiązanie należało stworzyć pętlę `while` która wykonuje potrzebną ilość iteracji do uzyskania 1 milimetrowej dokładności. Stworzono jeszcze "output" za pomocą `if`, `elif`, `else`, czyli to w jakiej chcemy aby funkcja zwracała nam wynik: stopnie dziesiętne, radiany lub stopnie minuty sekundy. Do tej ostatniej metody stworzyliśmy osobną funkcję `dms`. Postanowiono określić output funkcji dla wielofunkcyjności algorytmu. Wykorzystano również funkcję `get_np` która liczy promień przekroju w pierwszym wertykale. Wyniki zostały sprawdzone z wynikami jakie oddano w zaliczonym sprawozdaniu na semestrze III.

3.3 fh2XYZ

Transformacja fh2xyz przelicza współrzędne geodezyjne (ϕ , λ , H) na współrzędne kartezjańskie (x y z). Transformacja była używana na zajęciach z Geodezji wyższej. Wtedy poznano idee tego rozwiązania. Natomiast do przypomnienia skorzystano z strony www.asgeupos.pl - pdf z geodezyjnymi transformacjami. W funkcji implementującej to rozwiązanie należało zastosować trzy wzory. Każdy z nich odpowiada jednej współrzędnej kartezjańskiej. Wykorzystano również funkcję `get_np`. Wyniki zostały sprawdzone z wynikami jakie oddano w zaliczonym sprawozdaniu na semestrze III.

3.4 fl2PL1992

Transformacja fl2PL1992 przelicza współrzędne geodezyjne (ϕ , λ) do układu 1992 (x_{1992} y_{1992}). Transformacja była używana na zajęciach z Geodezji wyższej. Wtedy poznano idee tego rozwiązania. Natomiast do przypomnienia skorzystano z strony www.asgeupos.pl - pdf z geodezyjnymi transformacjami. W funkcji implementującej dodaliśmy warunek na ϕ i λ tak aby obejmowały tylko teren Polski. W przeciwnym razie wyskoczy błąd. Następnie zostały policzone poszczególne wartości z odpowiednich wzorów. Na końcu otrzymaliśmy x_{1992} oraz y_{1992} . Wyniki zostały sprawdzone z wynikami jakie oddano w zaliczonym sprawozdaniu na semestrze III.

3.5 fl2PL2000

Transformacja fl2PL2000 przelicza współrzędne geodezyjne (ϕ , λ) do układu 2000 (x_{2000} y_{2000}). Transformacja była używana na zajęciach z Geodezji wyższej. Wtedy poznano idee tego rozwiązania. Natomiast do przypomnienia skorzystano z strony www.asgeupos.pl - pdf z geodezyjnymi transformacjami. W funkcji implementującej przebiega prawie identycznie jak transformacja fl2PL1992. Zmieniona została skala oraz doczyt która ze stref 5,6,7,8 jest dla strefą dla naszych danych. Na końcu otrzymaliśmy x_{2000} oraz y_{2000} . Wyniki zostały sprawdzone z wynikami jakie oddano w zaliczonym sprawozdaniu na semestrze III.

3.6 xyz2neu

Transformacja fl2PL2000 przelicza współrzędne kartezjańskie (x y z) do układu neu. Transformacja była używana na zajęciach z Geodezji wyższej. Wtedy poznano idee tego rozwiązania. Natomiast do przypomnienia skorzystano z strony www.asgeupos.pl - pdf z geodezyjnymi transformacjami. W tym celu stworzono trzy definicje. Pierwszą na macierz obrotu (`renu`). Drugą liczy macierz zawierającą różnice między dwoma punktami, a trzecia liczy już macierz neu w której pierwsza kolumna liczy n druga e , a trzecia u . Wyniki zostały sprawdzone z wynikami jakie oddano w zaliczonym sprawozdaniu na semestrze III.

3.7 Wczytywanie i zapisywanie pliku

W celu odczytu i zapisu pliku zrobiono trzy funkcje. Pierwsza odczytuje plik txt Druga transformuje zmienne głównie do stringów. Robione jest to dla tego by wszystkie zmienne w pliku miały tą samą długość. Do tej zamiany użyto funkcji takich jak:

- `zamianan_float2string`
- `zamianan_float2string_fl`
- `zamianan_float2string_rad`

Wszystkie działają na takiej zasadzie, czyli za pomocą pętli while dodawana jest spacja przed liczbą. Warunek się kończy wtedy kiedy string będzie miał odpowiednią ilość znaków. Trzecia zapisuje plik w postaci tabelki z nagłówkiem. Postanowiono że dla punktu nienależącego do polskie w notatniku przy wyniku dla (x_{1992} y_{1992} x_{2000} y_{2000}) zostaną zapisane myślniki (-)

3.8 Testy dla funkcji

Dla pewności użytkownika czy program nie doznał niechcącej zmiany stworzyliśmy plik o nazwie `Testy_dla_funkcji.py`. Użyto biblioteki `pytest` która nie jest wbudowany w pythona. Należy go zainstalować w command window przy użyciu `pipa`. Przy użyciu `anacondy` instalacja nie jest potrzebna. Za pomocą komendy `assert` sprawdzane są wyniki pozyskane z zaliczonych sprawozdań z geodezji wyższej na semestrze III.

3.9 Kalkulatory transformacji i zapis ich wyników do Kalkulatora

W celu pojedynczej transformacji współrzędnych do innych układów stworzono 4 kalkulatory. Pierwszy z nich nosi nazwie **"Kalkulator_xyz2flh_PL1992_PL2000"** i wykorzystuje zimportowane transformacje z pliku głównego oraz bibliotekę `argparse`, gdzie skorzystano z `ArgumentParser`. Ma to na celu możliwość podawania przez użytkownika współrzędnych kartezjańskich (x y z) oraz elipsoidy. Następnie są liczone do układu geodezyjnego (ϕ , λ , H) oraz w układzie PL1992 i PL2000. Na tej samej zasadzie powstały kolejne kalkulatory:

- **"Kalkulator_xyz2neu"** - liczy z współrzędnych kartezjańskich (x y z) do układu neu
- **"Kalkulator_flh2xyz_PL1992_PL2000"** - liczy z współrzędnych geodezyjnych (ϕ , λ , H) do układu kartezjańskiego (x y z), PL1992 i PL2000.
- **"Kalkulator_flh2neu"** - liczy z współrzędnych geodezyjnych (ϕ , λ , H) do układu neu
- **"Kalkulator_xyz2flh_PL1992_PL2000"** - liczy z współrzędnych kartezjańskich (x y z) do układu geodezyjnego (ϕ , λ , H), PL1992 i PL2000.

Stworzono specjalne funkcje zapisujące wyniki w pliku głównym tj. `Transformacje_Projekt.py`. Jedna z nich zapisuje wyniki `xyz_flh_PL1992_PL2000`, a druga `neu`. W nich przeprowadzono niezbędne operacje na zmiennych np. zamienienie z float na str przy zachowaniu odpowiedniej ilości miejsc. Wszystko po to aby podany plik wynikowy miał ładną postać. Również użyto biblioteki `os` aby rozpoznawać czy dany plik jest już na naszym komputerze. Powoduje to iż nie jest tworzony nowy dokument z nagłówkami tylko do danego dokumentu podawane są kolejne liniki.

3.10 Dodanie możliwości wczytania pliku w argparse

W głównym pliku znajduje się również możliwość wczytania i zapisania pliku wynikowego przez bibliotekę `argparse`. Czyli dodano możliwość wczytania pliku z cmd. Korzystano z funkcji opisanych w rozdziale 3.7. Należy pamiętać że wczytywany plik powinien mieć odpowiednią formę taką jak plik `"wsp_inp.txt"`, który znajduje się na zdalnym repozytorium GitHub

3.11 Dodanie graficznego interfejsu GUI

Za pomocą biblioteki `tkinter` stworzono dwa okna. Pierwsze z nich główne, a drugie wynikowe. Komenda `tk.Label()` służy do stworzenia tekstów ułatwiających użytkownikowi wprowadzania danych. Samo wprowadzenia danych odbyło się za pośrednictwem `tk.Entry`. Za pomocą komendy `.insert` dodano przykładowe wprowadzenie danych do okna. Okienka kalkulatora wyświetlają się nad wszystkimi aplikacjami które użytkownik ma włączone. W drugim oknie wyświetlane są wyniki.

4 Podsumowanie

4.1 Rezultaty

Link do zdalnego repozytorium GitHub: https://github.com/Sawoboh/Informatyka_Projek_1.git

Znajduje się na nim pliki o nazwie:

- Transformacje_Projekt.py - główny plik w którym znajdują się transformacje oraz wywołanie przykładowego pliku txt wraz z zapisem do pliku o nazwie Wyniki transformacji.txt.
- wsp_inp.txt - zawierają przykładowe dane, które możemy przeliczyć.
- Kalkulator.py - Cztery plik importujący bibliotekę argparse. Za pomocą wiersza poleceń (cmd) należy podać dane do obliczeń.
- Testy_dla_funkcji.py - plik importujący bibliotekę pytest. Za pomocą wiersza poleceń (cmd) podać dane do obliczenia.
- TKinter.py - plik wywołuje kalkulator graficzny i okno wynikowe.

4.2 Umiejętności nabyte

- Sprawne pisanie plików tekstowych w latex w celu nauki skorzystał z książki (Borkowski i Przybylski, 2015)
- Pisanie kodu obiektowego w Pythonie
- Posługiwanie się bibliotekami takimi jak: argparse, pytest, tkinter, os, numpy (Chaniewski, 2018) (Stevenson, 2021)
- Praca zespołowa z wykorzystaniem platformy Github
- Poprawienie jakości i przyspieszenie pisania kodu w Pythonie

4.3 Spostrzerzenia, problemy i ich rozwiązania:

Spostrzerzenia:

- Wraz ze wzrostem ilości czasu poświęconego na projekt, zauważano coraz więcej luk w funkcjach, które usprawniono.
- Cały czas program nie jest kompletny w 100%. Zawsze się znajdzie nowy pomysł który można zaimplementować.

Problem	Rozwizanie
Brak spójności w długości liczb w tabeli przez co plik wynikowy nie wyglądałby schludnie	Dodanie funkcji na dodanie spacji w stringu opisany w rozdziale 3.7
Pojawianie się błędu w funkcji PL2000 i PL1992 dla (ϕ, λ) nie leżącego na terenie Polski	Dodanie warunku na (ϕ, λ) w którym jeśli warunek nie jest spełniony zapisuje w notatniku myślinik (-)
Dopisywanie do istniejącego pliku wyników	Wykorzystano bibliotekę os do rozpoznania czy dany plik istnieje i podanie odpowiednich komend w przypadku gdy plik zostaje stworzony i do pliku zostają dopisane wyniki
Nieumiejętność korzystania z biblioteki argparse	Dopytywanie się na zajęciach informatyki geodezyjnej prowadzącego o wytłumaczenie jak korzystać z biblioteki
Nieumiejętność korzystania z portalu Github	Skorzystanie z materiałów uspołecznionych przez prowadzącego zajęcia informatyka geodezyjna
Brak możliwości zresetowania tablicy z wynikami w pliku pytest	

Literatura

- Borkowski, M., i Przybylski, B. (2015). *Książka kucharska latex*. Springer Wien New York.
- Chaniewski, P. (2018). *Python argparse - przekazywanie parametrów (argumentów) wiersza poleceń*. <https://cws.pl/python/tutorial/python-argparse-przekazywanie-parametrow-argumentow-uruchomieniowych-do-skryptu/?fbclid=IwAR0ctjScWQ55Jeyhvl7gVSP0lrGLeeKYItZFliZINUfwEhES1Bn00NEaf3E>.
- Kadaj, R. J. (2002). *Polskie układy współrzędnych polskie układy współrzędnych polskie układy współrzędnych formuły transformacyjne, algorytmy i programy*. http://www.geonet.net.pl/images/2002_12_uklady_wspolrz.pdf.
- Stevenson, V. (2021). *Command Line Parsing Arguments in Python with Argparse - Intro and Demo*. https://www.youtube.com/watch?v=53H_082uqfY.
- Wang, P., i Oliphant, T. (2012). *Anaconda*. <https://www.anaconda.com/download>.