GIS support

Materiały szkoleniowe

Zaawansowany QGIS
(poziom zaawansowany)

Ministerstwo Klimatu i Środowiska

Sfinansowano ze środków Narodowego Funduszu Ochrony Środowiska i Gospodarki Wodnej

Spis treści:

Niniejszy zeszyt ćwiczeń został opracowany dla programu QGIS Frankfurt w wersji LTR opatrzonej numerem 3.16.7. Ponadto wykorzystano programy SAGA GIS w wersji 2.3.2. oraz GRASS GIS w wersji 7.8.5 Do ćwiczeń dołączono paczkę z danymi, zawierającą następujące warstwy przestrzenne:

- katalog CLC_2018
- GLB_VOLC.shp
- gminy.gpkg
- gminy_lubelskie.shp
- Linie.shp
- miejscowosci.shp
- parki_narodowe.gpkg
- powiaty.gpkg
- RYSY_DEM.tif
- wojewodztwa.gpkg

Zaawansowana edycja danych w QGIS. Wyrażenia warunkowe, agregujące i działanie na geometriach w Kalkulatorze Pól

Dane są dwie warstwy o rozszerzeniach .gpkg - powiaty i gminy. Pierwsza zawiera pełen wykaz krajowych powiatów wraz z geometriami reprezentującymi ich granice administracyjne, druga zaś gminy województwa wielkopolskiego. Celem jest wykonanie złączenia danych z obu warstw wyłącznie przy użyciu *Kalkulatora Pól*. W pierwszej kolejności przyłączymy informację o powiatach do gmin.

Otwieramy *Tabelę Atrybutów* warstwy *gminy* i przechodzimy do *Kalkulatora Pól*. Tworzymy nowe pole o nazwie *powiat*. Złączenia dokonujemy wykorzystując wyrażenie agregujące, uwzględniające częściową zbieżność kodów TERYT wedle zasady: kod TERYT powiatu = cztery pierwsze cyfry kodu TERYT gminy:

```
aggregate(
 layer:='powiaty',
 aggregate:='concatenate',
 expression:="powiat",
 concatenator:='',,
 filter:= left(attribute(@parent,'TERYT_gm'),4) =
 "TERYT_pow")
```

Rozłóżmy zapis na “czynniki pierwsze” i przeanalizujmy poszczególne partie. Funkcja aggregate() pozwala tworzyć połączenia z polami innych warstw na podstawie relacji między geometriami obiektów lub zawartości pól. Przyjmuje ona kilka argumentów. Argument pierwszy, “*layer:=*” jako wartość przyjmuje nazwę warstwy, do której się odwołujemy. Nazwa ta musi być ujęta w apostrofy. Argument drugi, “*aggregate:=*”, pozwala określić typ operacji łączenia. Wartość ‘concatenate’ pozwala na przyłączenie danych z innego pola. “*Expression:=*” to argument, w którym wskazujemy pole, którego zawartość zostanie przyłączona. “*Concatenator:=*” jest argumentem opcjonalnym, niezbędnym jednak w sytuacji, gdy pole naszej warstwy może zawierać więcej niż jedną wartość przyłączoną. Jako wartość przyjmuje separator - zwyczajowo w postaci przecinka. Wreszcie argument “*filter:=*” pozwala dookreślić kryteria samego złączenia. Docelowo wyrażenie filtrujące odwołuje się do geometrii i pól warstwy, której pola chcemy przyłączyć. Aby odwołać się do obiektów z warstwy macierzystej, należy zastosować zapis *@parent*.

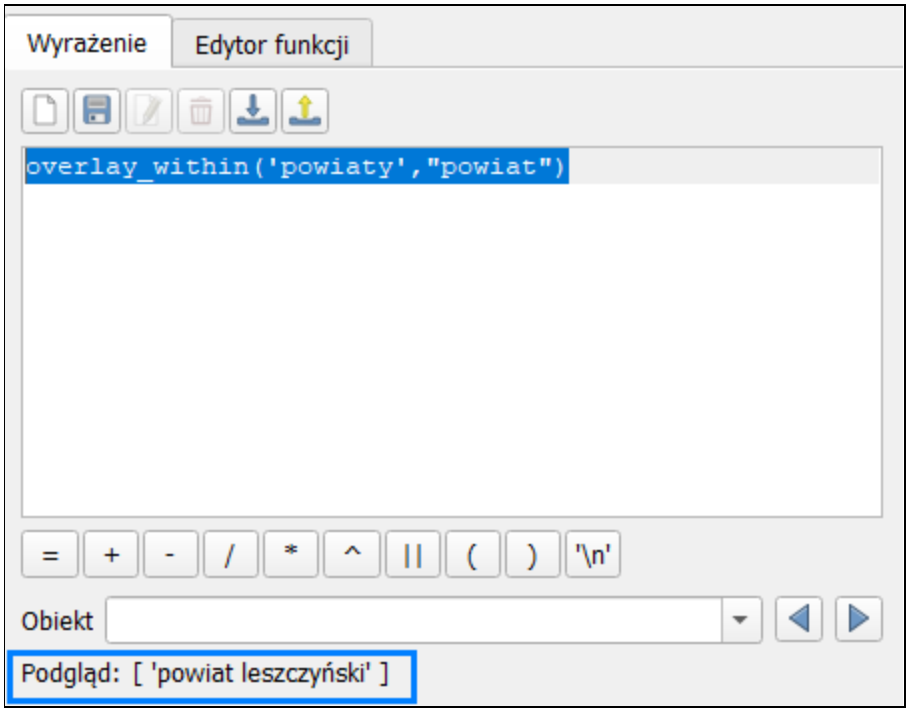
Złączenie można również wykonać wykorzystując fakt, że geometrie gmin powinny zawierać się ściśle w geometriach powiatów. W takim wypadku modyfikacji podlega wyłącznie zawartość argumentu *filter:=* :

```
aggregate(
layer:='powiaty',
aggregate:='concatenate',
expression:="powiat",
concatenator:='',,
filter:= within(geometry(@parent),$geometry))
```

Dodatkowo od wersji 3.16 wprowadzono funkcje *overlay_*, dzięki którym tworzenie agregacji jest jeszcze prostsze. Przykładowo, chcąc odtworzyć wcześniej opisane działanie skorzystamy z funkcji *overlay_within()*. Wymaga ona w zasadzie tylko jednego argumentu w postaci nazwy drugiej warstwy, niemniej wykorzystując argumenty dodatkowe możemy precyzyjnie wyznaczyć m.in. wartości konkretnych atrybutów:

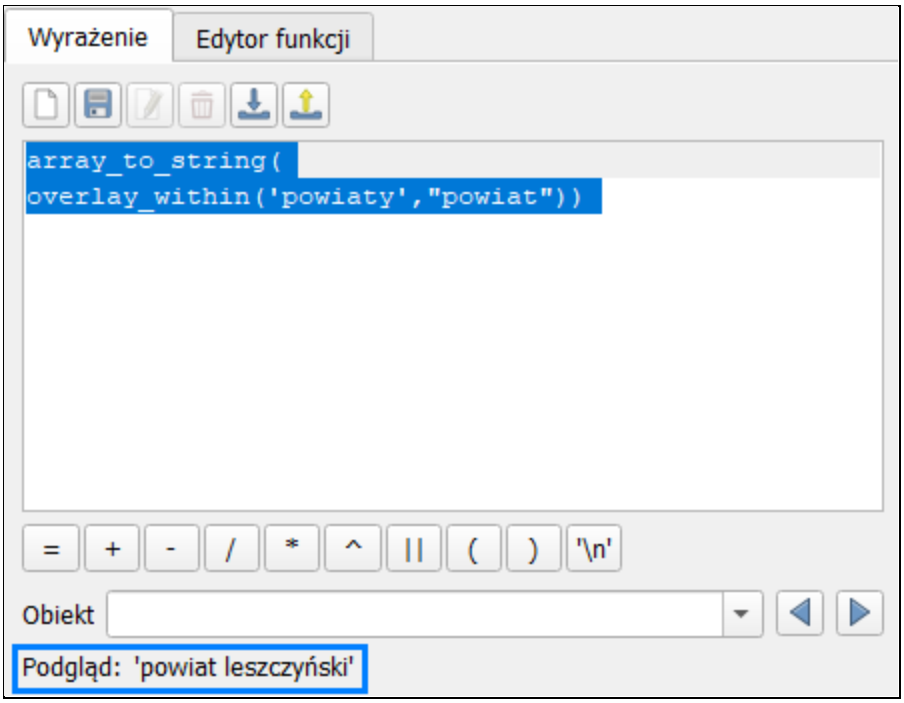
overlay_within('powiaty',"powiat")

Musimy jednak mieć na uwadze, że wynikiem działania nie jest wartość tekstowa, a lista wartości, o czym świadczy obecność nawiasów kwadratowych w podglądzie wyniku:



Aby wydobyć wartość opisową z listy, należy skorzystać z funkcji *array_to_string()*:

array_to_string(
overlay_within('powiaty',"powiat"))



Konkatenacja, czyli przyłączenie ciągu znaków ze wskazanej warstwy, to tylko jedna z wielu operacji, jakie możemy wykonać przy użyciu wyrażeń agregujących. Dodajmy do bieżącego projektu warstwę *miejscowości* i spróbujmy obliczyć, ile punktów znajduje się w granicach poszczególnych gmin:

```
aggregate(
layer:='msc',
aggregate:='count',
expression:=$id,
concatenator:='',
filter:= intersects($geometry,geometry(@parent)))
```

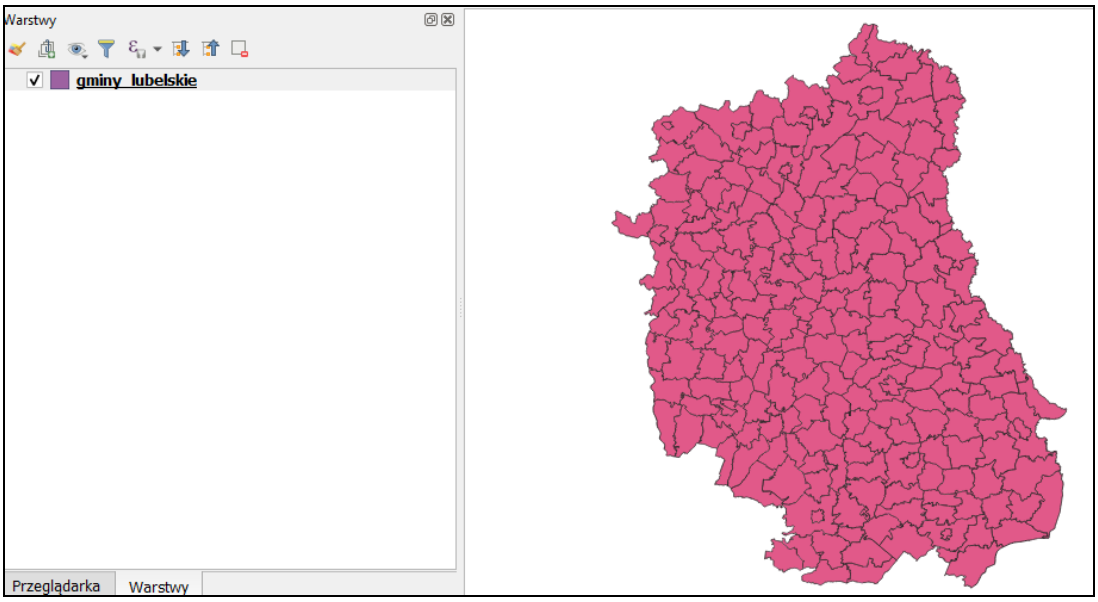
Wejdźmy na jeszcze wyższy poziom szczegółowości i sprawdźmy, jaka jest średnia długość nazw miejscowości w gminach woj. wielkopolskiego:

```
round(
aggregate(
layer:='msc',
aggregate:='mean',
expression:=length("nazwa"),
concatenator:='',
filter:= intersects($geometry,geometry(@parent))))
```

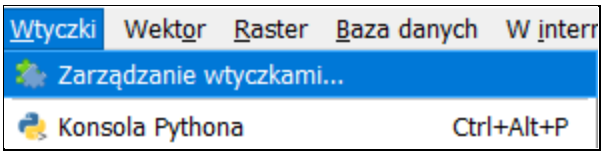
Sprawdzanie i korekta geometrii obiektów wektorowych z wykorzystaniem modułów QGIS i GRASS GIS

Celem ćwiczenia jest zapoznanie uczestników z narzędziami kontroli topologii oraz sprawdzania geometrii warstw wektorowych. Program QGIS udostępnia użytkownikom rozwiązania natywne, jak również wybrane moduły takich systemów jak GRASS GIS czy SAGA GIS. Błędy w geometrii dotyczą obiektów, które taką geometrię posiadają, a więc jedno- (linie i polilinie) i dwuwymiarowych (wieloboki). Do najczęstszych błędów o charakterze topologicznym należą powtarzające się geometrie (duplikaty), samoprzecięcia, niedociągnięte wierzchołki lub “szczeliny” w przebiegu wspólnej granicy.

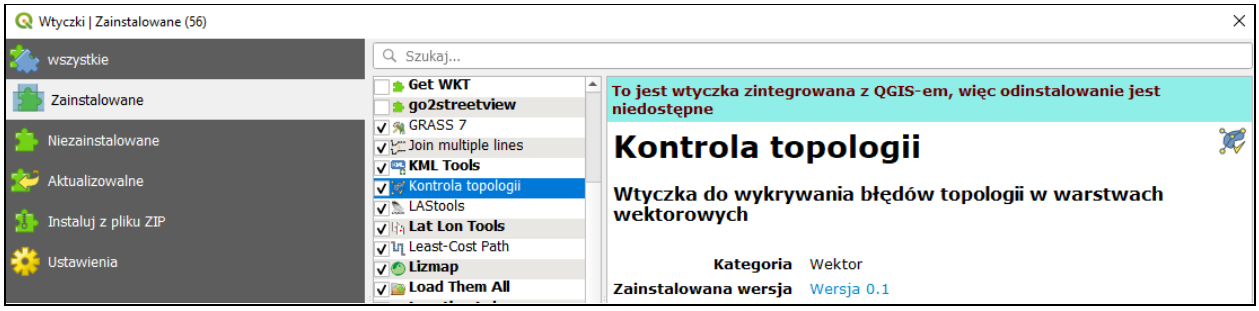
Zacznijmy od utworzenia nowego projektu w układzie współrzędnych PUWG 1992 oznaczonym kodem EPSG:2180. Następnie dodajmy do widoku mapy warstwę .shp *gminy_lubelskie*.



Sprawdźmy warstwę pod kątem poprawności topologicznej. W szczególności interesuje nas obecność duplikatów lub wspomnianych przerw między sąsiadującymi jednostkami administracyjnymi. Do sprawdzania topologii użyjemy natywnej wtyczki o nazwie *Kontrola Topologii*. UWAGA! Wtyczka powinna być aktywna od razu po instalacji programu QGIS, niemniej stan ten nie stanowi reguły. Zaczniemy więc od sprawdzenia stanu wtyczek w zakładce *Wtyczki -> Zarządzanie wtyczkami*:

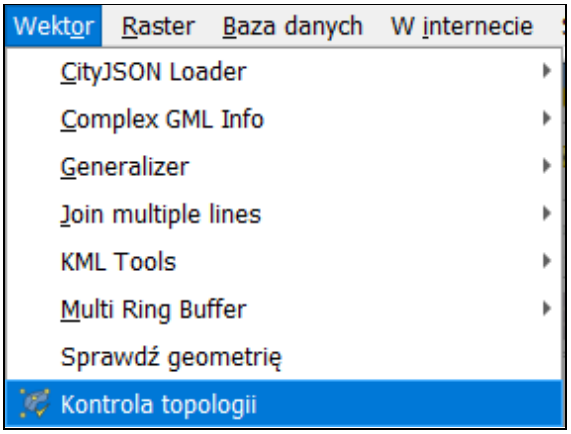


Po otwarciu okna przechodzimy do zakładki *Zainstalowane* i lokalizujemy pozycję *Kontrola Topologii* upewniając się, że checkbox przy nazwie narzędzia jest zaznaczony:

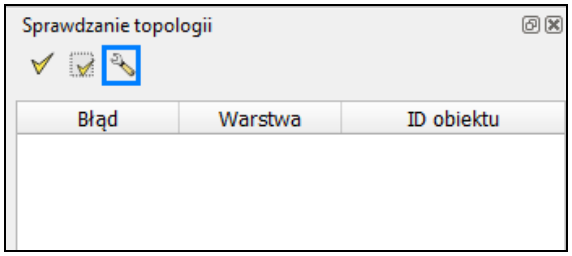


W tym samym miejscu możemy dodatkowo sprawdzić, czy posiadamy aktywną wtyczkę *Sprawdź geometrie*. W innym wypadku należy kliknąć checkbox przy jej nazwie:

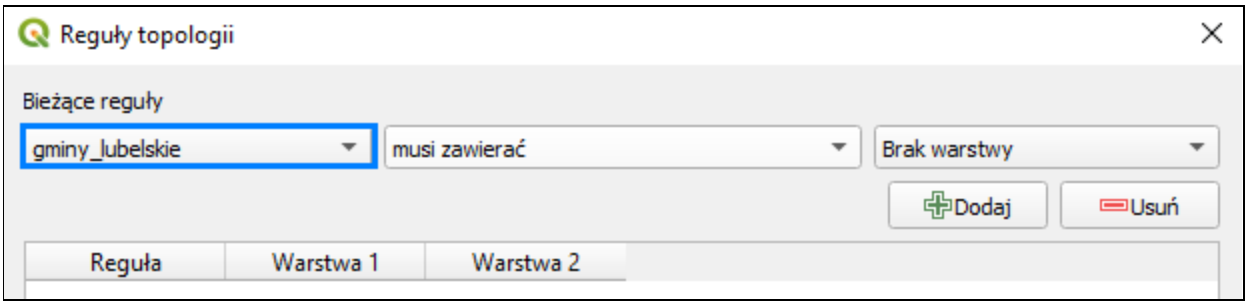
Po wprowadzeniu niezbędnych zmian możemy zamknąć okno wtyczek i wrócić do widoku mapy. Aby przejść do kontroli topologii należy otworzyć panel rozwinąć zakładkę *Wektor* z pasku *menu* i wybrać odpowiednią opcję:



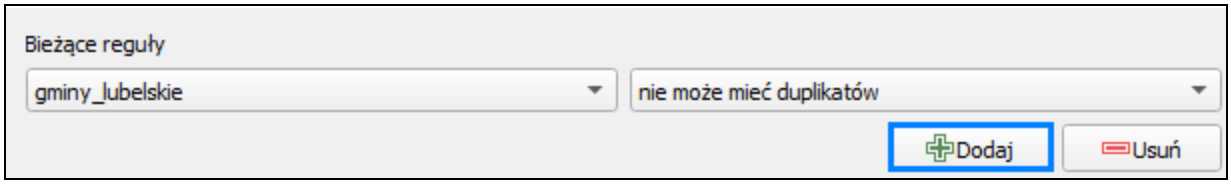
Po uruchomieniu okna narzędzia należy przejść do ustawień konfiguracji. W tym celu klikamy na jedną z ikon na pasku narzędzi w górnej części aktywnego okna:



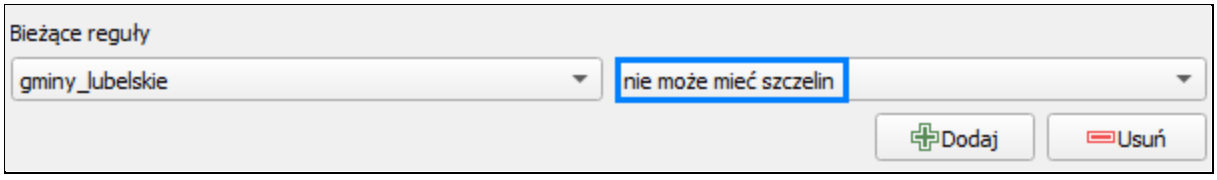
W kolejnym oknie mamy możliwość wskazania warstwy, której dotyczy kontrola, oraz określenia reguł branych pod uwagę w trakcie sprawdzania poprawności topologii. Warstwę wskazujemy z rozwijanej listy po lewej stronie okna:



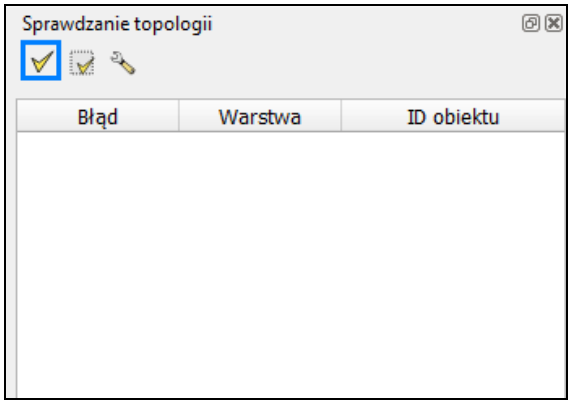
Reguły natomiast wybieramy w oknie sąsiednim. Przykładowo chcemy sprawdzić, czy warstwa nie zawiera zduplikowanych geometrii. W tym celu w środkowym oknie należy wybrać opcję *"nie może mieć duplikatów"*. Regułę dodajemy do listy klikając na *Dodaj*:



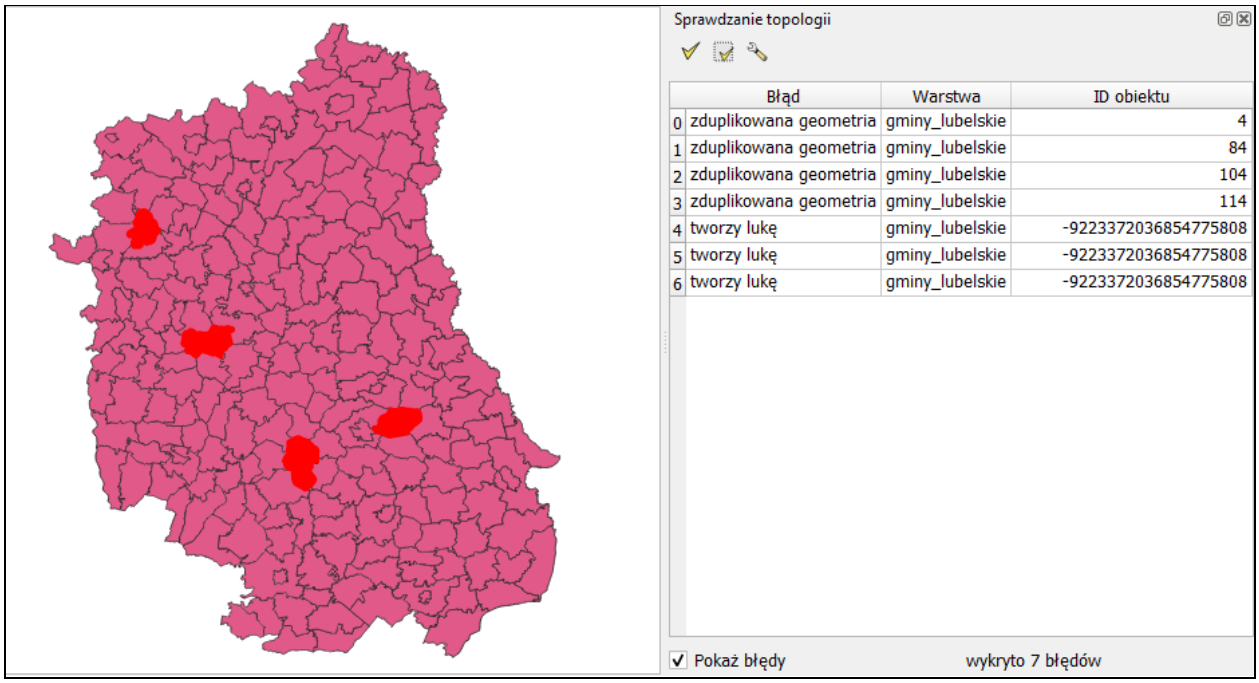
W analogiczny sposób możemy dodać regułę sprawdzającą warstwę pod kątem obecności szczelin:



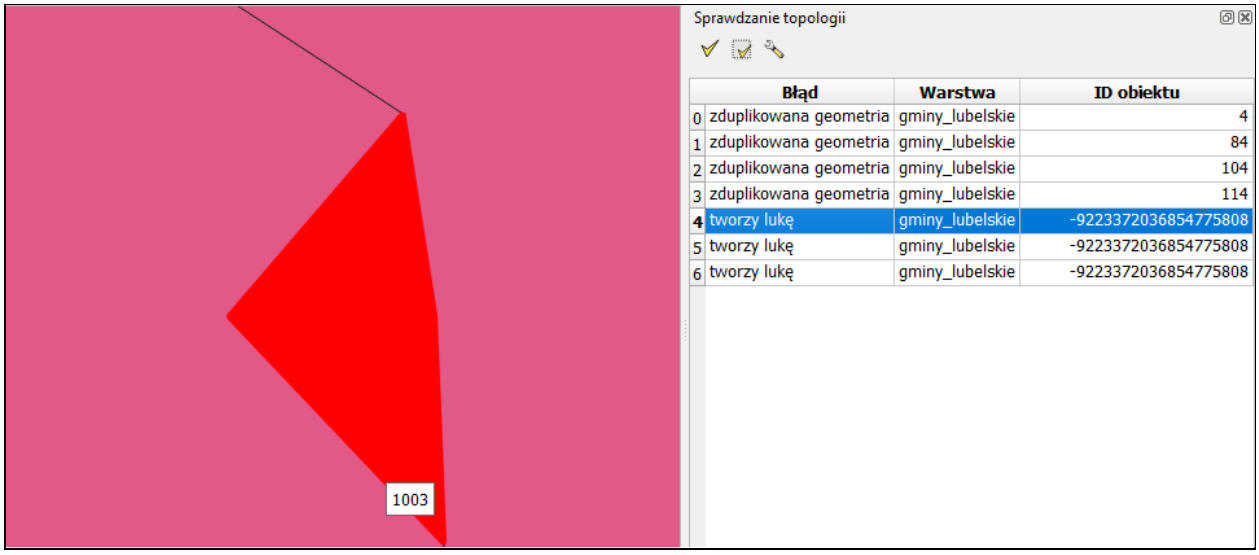
Po wybraniu reguł klikamy na *OK* i wracamy do poprzedniego widoku. Aby aktywować proces sprawdzania, klikamy na “ptaszek”:



Po chwili okno wypełni lista błędów wraz opisem. Dodatkowo lokalizacje poszczególnych nieścisłości topologicznych oznaczone są kolorem czerwonym w oknie mapy:

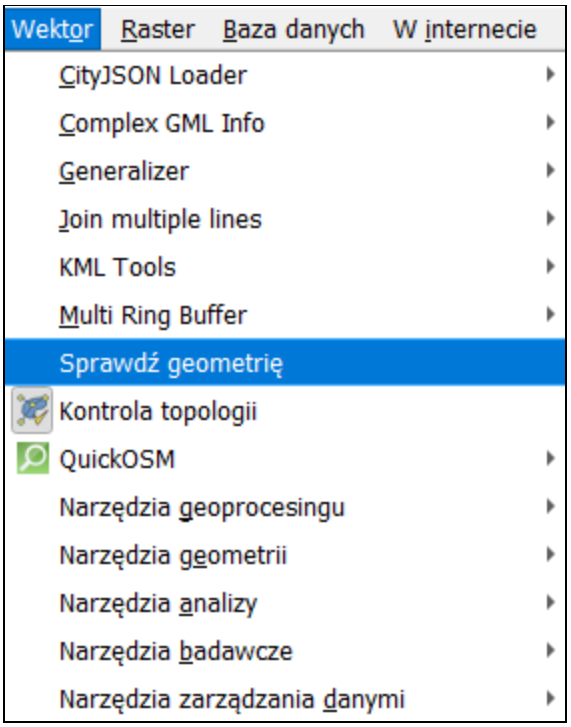


Ponadto istnieje możliwość bliższego przyjrzenia się poszczególnym błędom. W tym celu wystarczy pojedyncze kliknięcie na rekord zawierający opis interesująco nas błędu. W ten sposób wyśrodkujemy widok okna mapy na wskazanym obszarze:

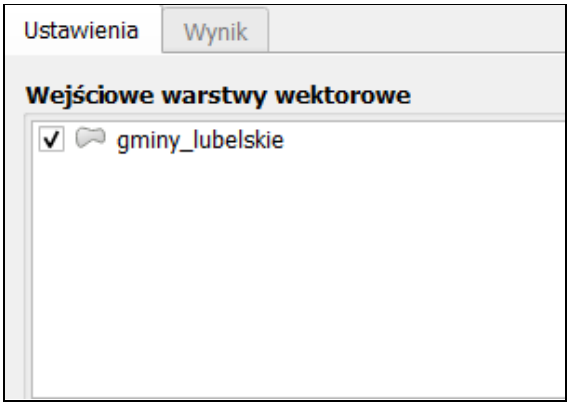


Znając lokalizację poszczególnych błędów w topologii, możemy dokonać ich ręcznej korekty. Oczywiście taki tryb pracy dotyczy jedynie sytuacji, w której liczba błędów jest relatywnie mała. Co jednak, jeśli liczba obiektów wymagających korekty sięga setek lub tysięcy? W podobnych przypadkach jedynym rozwiązaniem pozostają narzędzia automatycznej korekty. Należy przy tym zaznaczyć, że nie zawsze można przy ich pomocy rozwiązać wszystkie wykryte problemy. Ponadto nierzadko zdarza się, że proponowane przez system rozwiązanie wydaje nam się niewłaściwe. Jednym ze sposobów radzenia sobie z przywołanymi ograniczeniami jest stosowanie wybranych modułów do rozwiązywania konkretnych problemów z geometriami. Wrócimy do tego zagadnienia na etapie korekty warstwy liniowej.

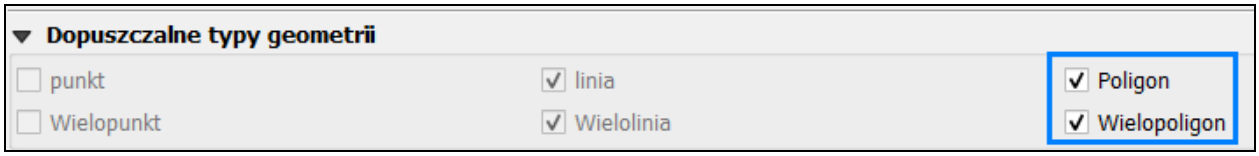
Aktualnie rozpatrywany przypadek zawiera natomiast standardowe błędy, z którymi QGIS powinien sobie poradzić. Do ich naprawy użyjemy narzędzia *Sprawdź geometrie*, znajdującego się w zakładce *Wektor*:



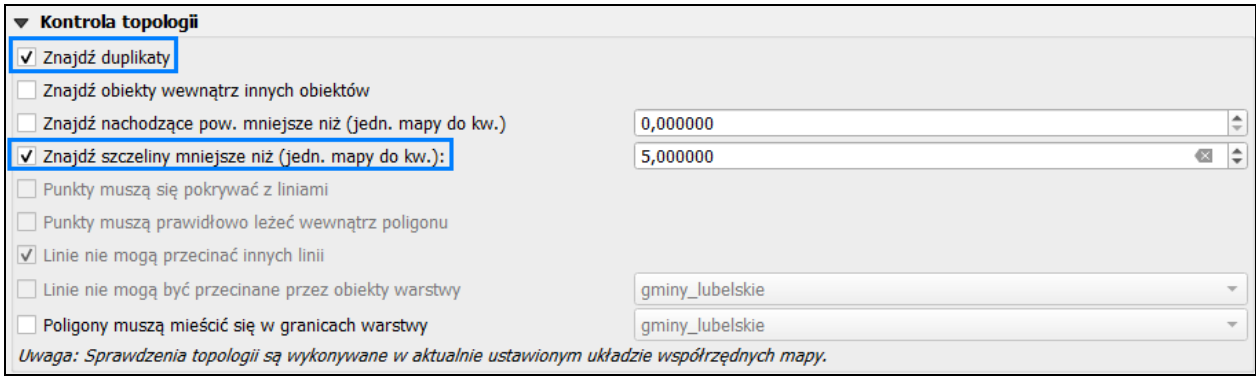
Po uruchomieniu wyświetli się okno dialogowe ze znaczną liczbą opcji do wyboru. Nasze bieżące działanie dotyczy konkretnej warstwy i typu geometrii. Zaczynamy więc od wskazania warstwy docelowej:



Zauważmy, że po wskazaniu warstwy do sprawdzenia program automatycznie rozpoznaje typ geometrii:

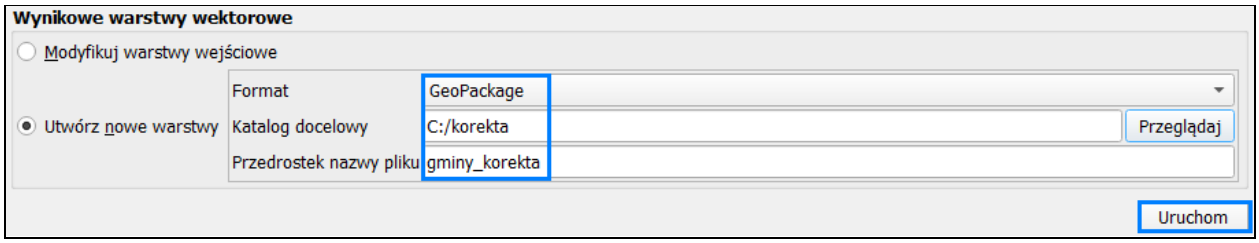


Na tej podstawie filtruje również listę kolejnych opcji. Przejdźmy do ustawień kontroli topologii:

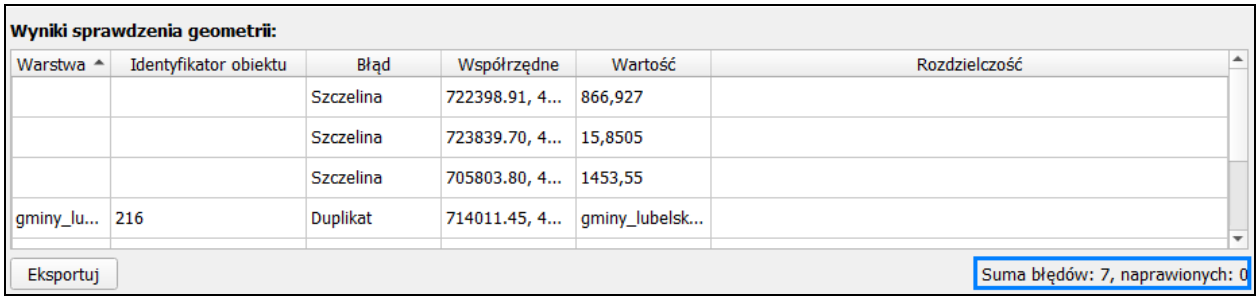


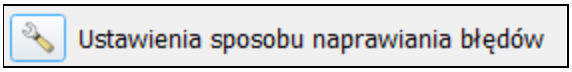
Pamiętamy, że nasz zbiór zawiera duplikaty oraz szczeliny. W przypadku tych drugich możemy dodatkowo ustawić próg określający wielkość kwalifikującą szczelinę do usunięcia. Domyślnie jest to 5 m kw. (w przypadku warstw o układach współrzędnych prostokątnych płaskich). **Jako że działamy w stosunkowo małej skali, warto zwiększyć tę wartość co najmniej do 10000**.

W oknie *Wynikowe warstwy wektorowe* możemy określić format warstwy wynikowej oraz zdefiniować ścieżki zapisu pliku. Ponadto istnieje możliwość modyfikacji geometrii źródłowej (z której w tym przypadku zrezygnujemy). Warstwę z poprawionymi geometriami zapisujemy w nowej geopaczce (nie ma potrzeby podawania nazwy). Wskazujemy również lokalizację pliku wynikowego - może to być np. nowy katalog o nazwie *C:\korekta*. Po ustawieniu poszczególnych parametrów klikamy na *Uruchom*.

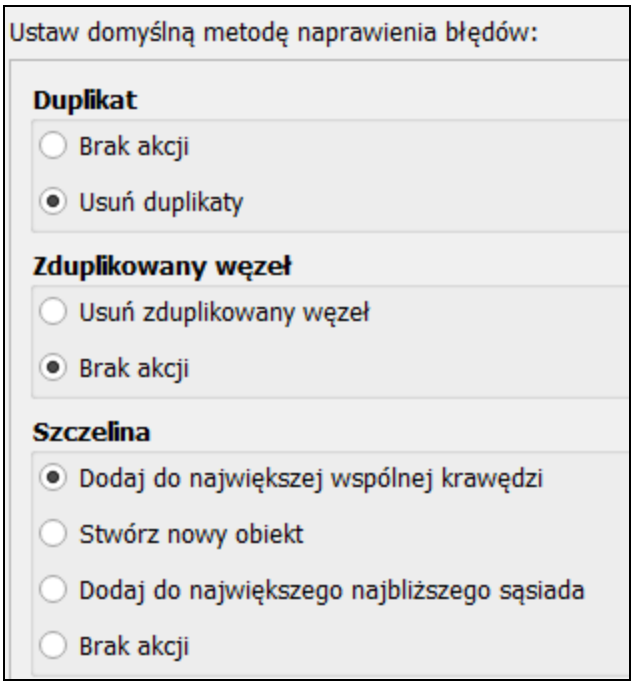


Po chwili oczekiwania wyświetli się kolejny ekran z listą wykrytych błędów. Jeśli dobrze dobraliśmy wartości parametrów kontroli, powinna ona liczyć tyle samo (7) pozycji co lista z narzędzia *Kontrola topologii*.



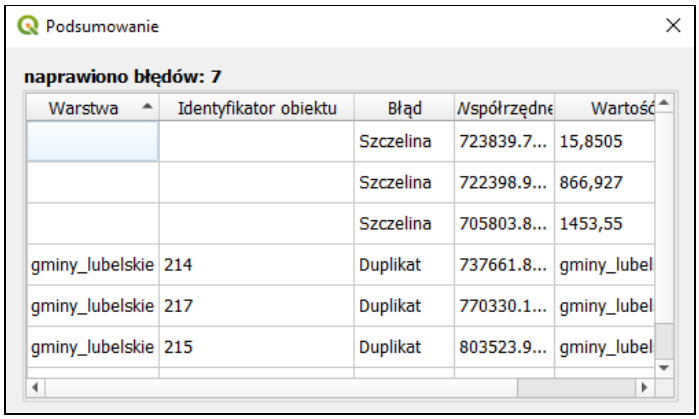
Różnica między obiema wtyczkami polega na tym, że *Sprawdź geometrię* pozwala w sposób zautomatyzowany naprawić/usunąć błędy wykorzystując domyślne lub wskazane przez użytkownika metody. Aby samodzielnie wskazać metody korekty, należy kliknąć na 

. Wyświetli się nowe okno zawierające wykaz kategorii problemów wraz z opcjami rozwiązań. Dla duplikatów wybierzmy opcję *Usuń*. W przypadku szczelin zaleca się skorzystanie z opcji *Dodaj do największej wspólnej krawędzi*:



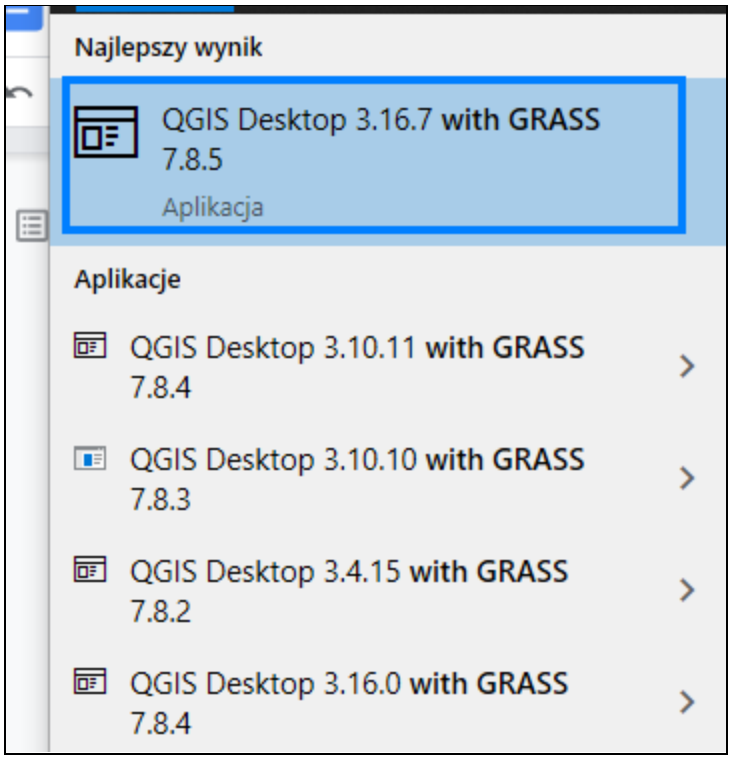
Po ustanowieniu metod klikamy na *OK*, następnie zaś na 

. Po zakończeniu przetwarzania program wyświetli nowe okno z podsumowaniem operacji:

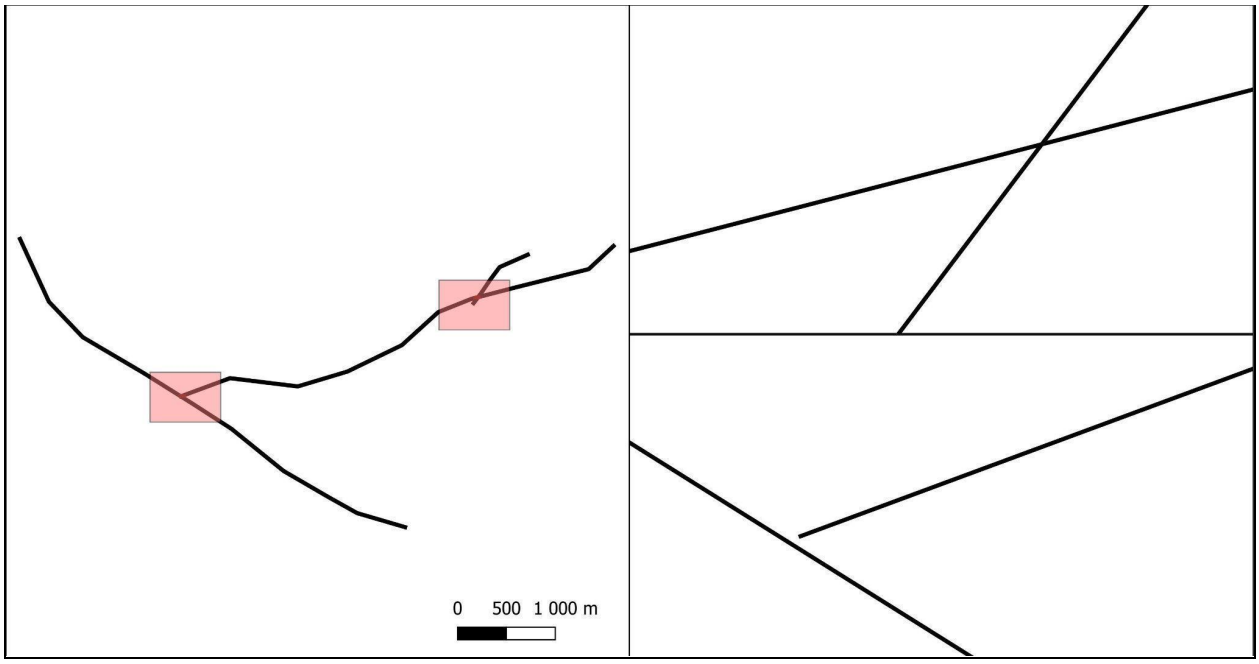


Naprawa błędów w geometrii/topologii z wykorzystaniem narzędzi GRASS GIS

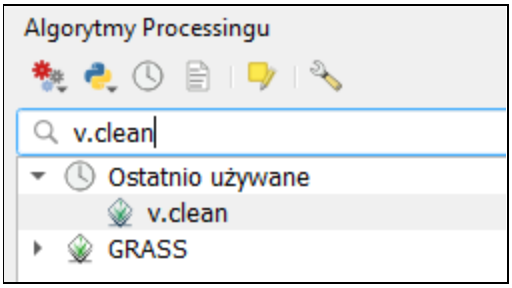
W kolejnym ćwiczeniu spróbujemy skorygować błędy w geometrii warstwy liniowej. Do wykonania zadania niezbędne będzie uruchomienie programu QGIS wraz z obsługą modułów GRASS GIS. Aby wyszukać i uruchomić odpowiednią wersję należy otworzyć menu *Start* systemu *Windows* i wpisać “with grass”. Na liście wyświetli się lista dostępnych opcji; wybieramy wersję 3.16 LTR z aktywną obsługą modułów:



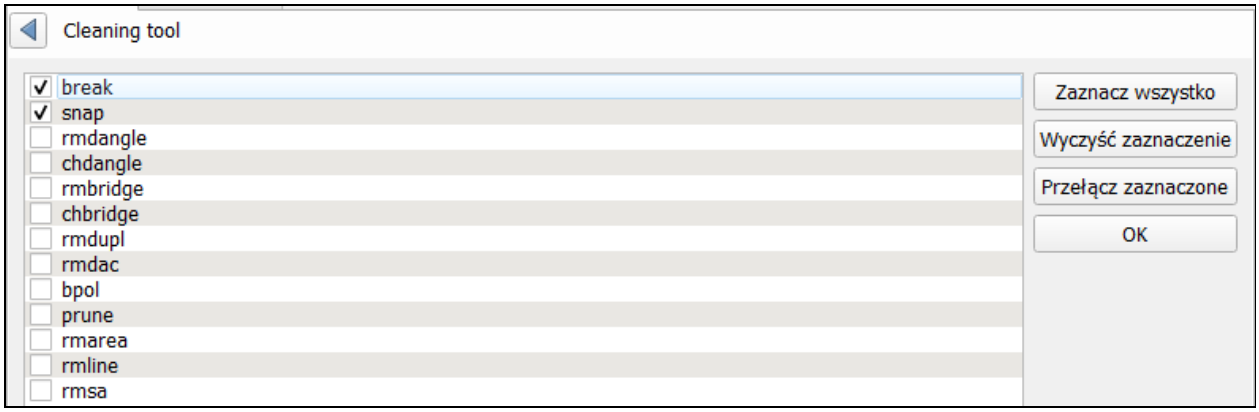
Po uruchomieniu programu pozostajemy w ramach nowego projektu. Nadajmy mu układ współrzędnych o kodzie EPSG:2180. Następnie dodajemy do projektu warstwę *Linie.shp*. Zawiera ona dwa znaczne błędy: niedociągnięty segment oraz skrzyżowanie linii:



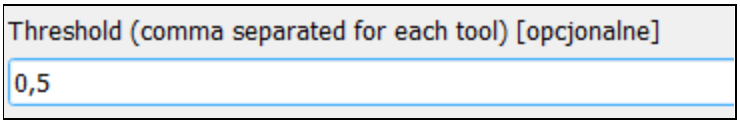
Do rozwiązania obu problemów spróbujemy wykorzystać moduł systemu GRASS o nazwie *v.clean*. Odnajdziemy go w oknie *Algorytmów Processingu* - wystarczy podać jego nazwę w polu *Szukaj*:



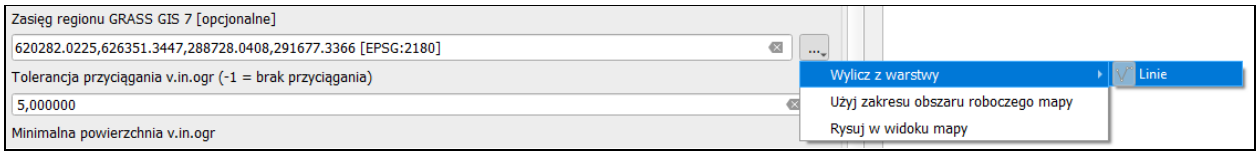
Uruchamiamy moduł. Zaczynamy od podania warstwy wejściowej - będą to *Linie*. Następnie przechodzimy do wyboru narzędzi naprawczych. Klikamy na *Cleaning Tool* i zaznaczamy dwie pierwsze opcje:



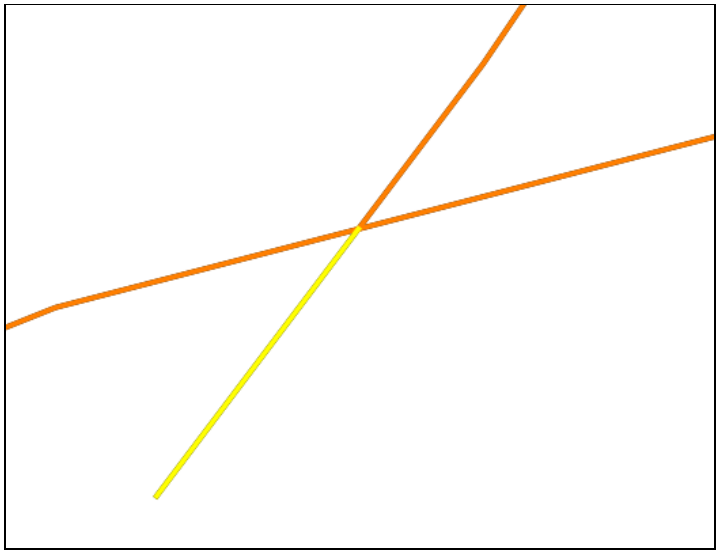
Wracamy do poprzedniego widoku, klikając na niebieską strzałkę w górnym, lewym rogu ekranu. Dodatkowo określamy wartości progowe dla obu narzędzi:

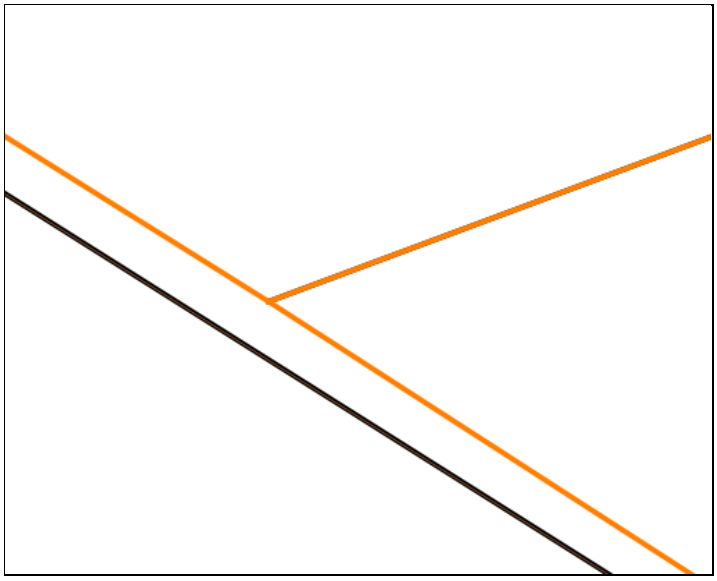


Z kolei w parametrach zaawansowanych wskazujemy zasięg obszaru (wyliczony z warstwy *Linie*).

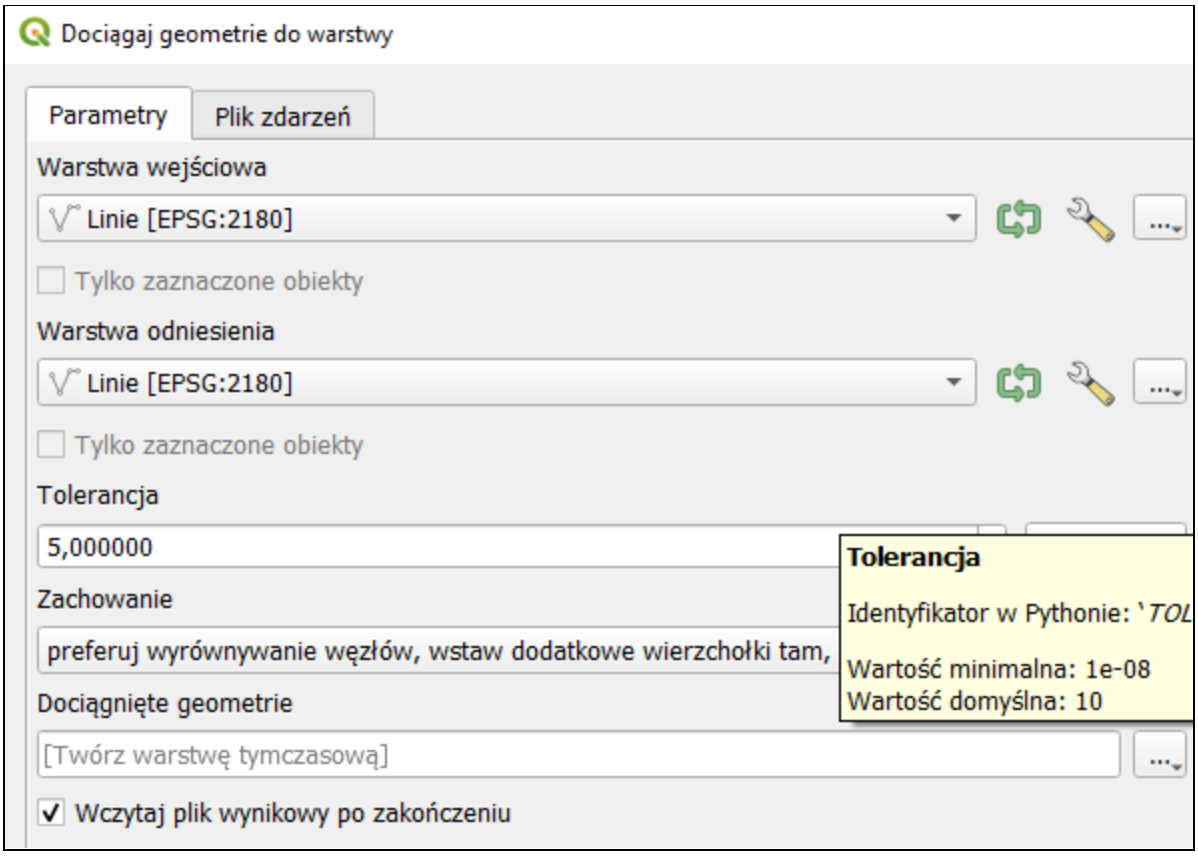


Klikamy na *Uruchom*. W oknie mapy możemy sprawdzić sposób rozwiązania obu problemów.

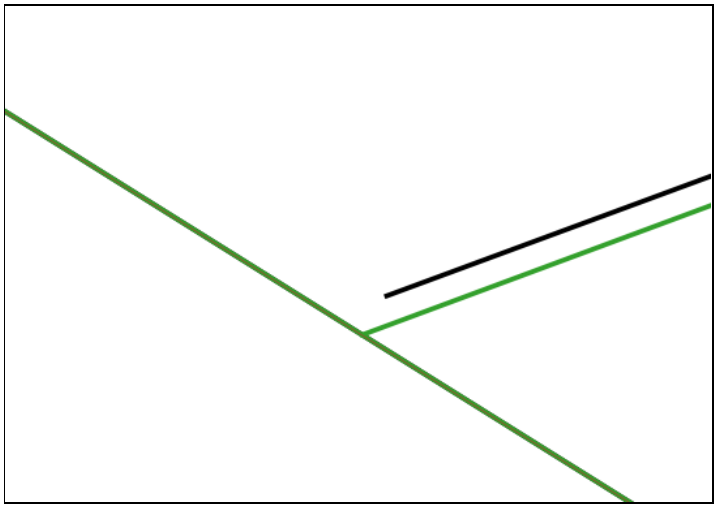




Program poradził sobie z przecięciem linii rozbijając geometrię jednego z segmentów na dwie niezależne części. W przypadku niedociągniętego wierzchołka zmienił jednak lokalizację całego sąsiedniego segmentu. Takie rozwiązanie problemu nie jest w pełni satysfakcjonujące. Oczywiście moglibyśmy metodą prób i błędów rozpoznać odpowiednią wartość i parametry korekty, niemniej często kluczowym czynnikiem analizy jest czas. W podobnej sytuacji można sięgnąć po szybsze i prostsze rozwiązania, jak chociażby narzędzie *Dociągaj geometrie do warstwy*. Można je znaleźć w panelu *Algorytmów Processingu*. Po uruchomieniu algorytmu jako warstwę wejściową ponownie wskazujemy *Linie*. W tym przypadku będzie ona dodatkowo pełnić funkcję warstwy odniesienia. Wielkość przyciągania określamy na 5 (wierzchołek jest bowiem oddalony od najbliższego segmentu o niespełna 2 metry). W polu *Zachowanie* pozostawiamy wartość domyślną:

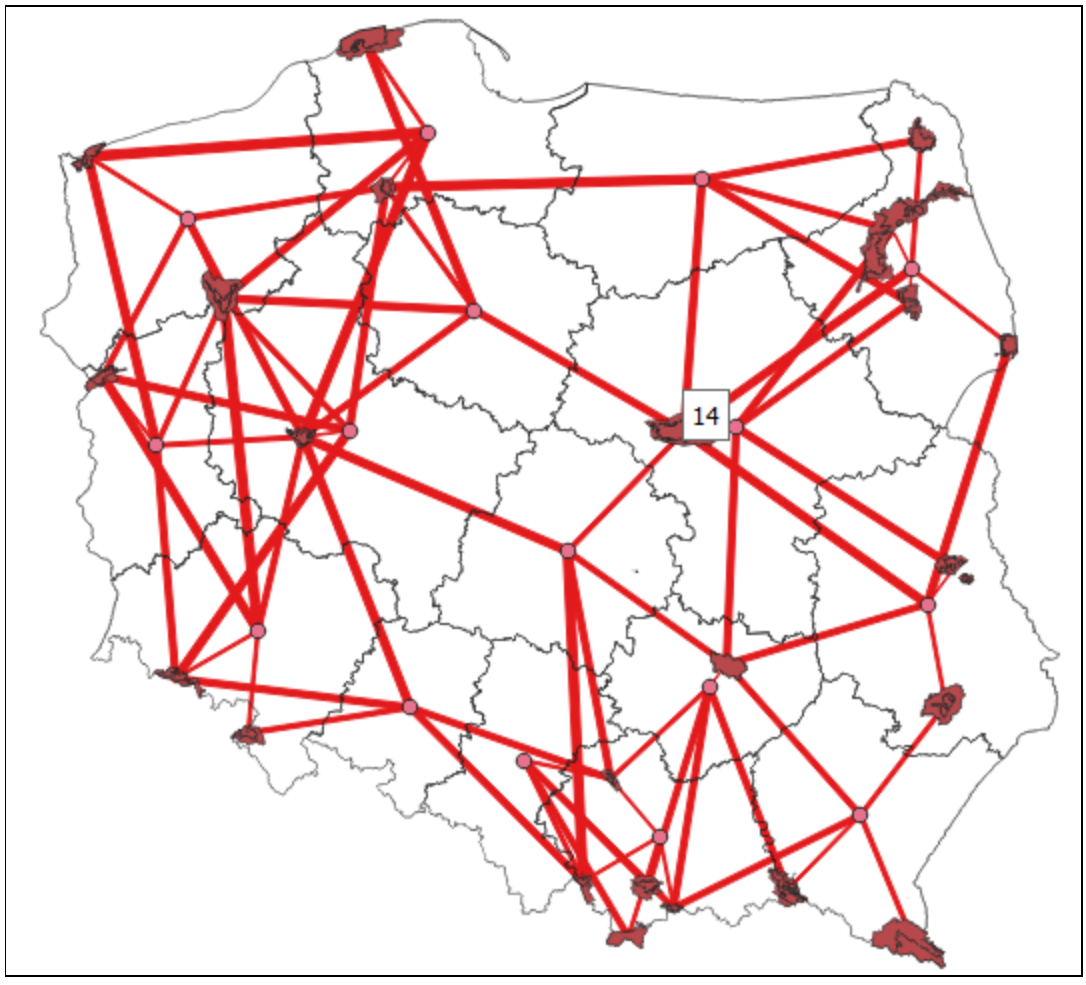


Klikamy na *Uruchom* i sprawdzamy wygląd warstwy wynikowej w oknie mapy. Tym razem geometria “odszczepionego” segmentu została przyłączona poprawnie (linię wynikową oznaczono kolorem zielonym).



Zaawansowana wizualizacja danych. Wykorzystywanie danych tekstowych i liczbowych do przygotowania dynamicznej symbolizacji warstw wektorowych

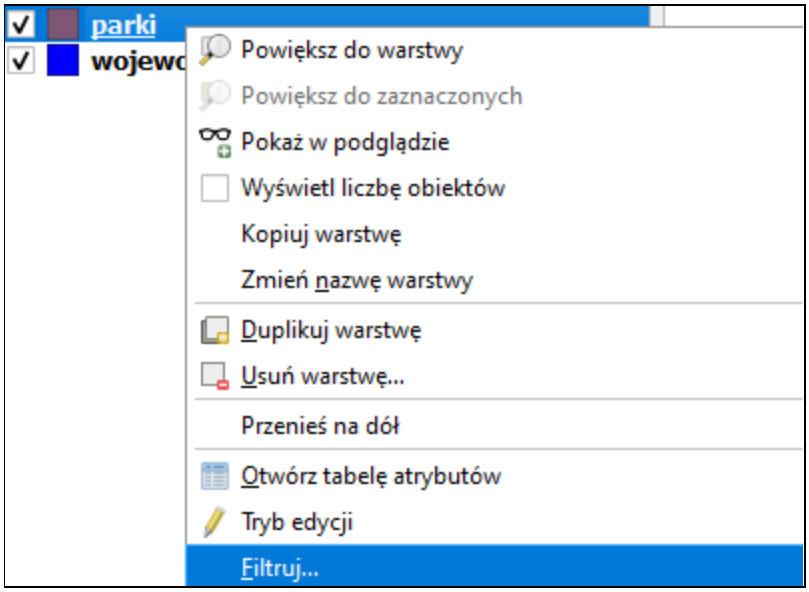
Celem zadania jest przygotowanie dynamicznej symbolizacji warstw, obrazującej najkrótsze połączenia liniowe między centroidami województw a pięcioma najbliższymi parkami narodowymi. Grubość linii łączącej punkty ma być wprost proporcjonalna do dzielącej je odległości. Efekt końcowy powinien być zbliżony do przedstawionego poniżej:



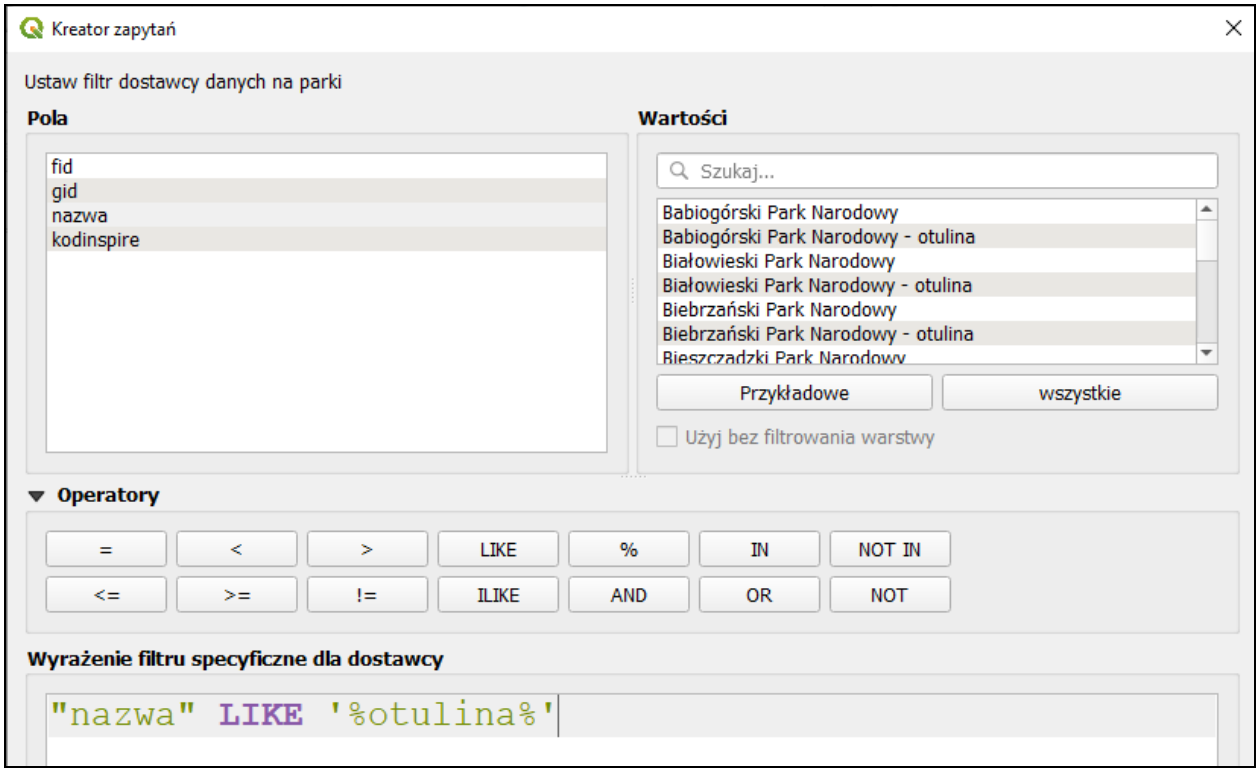
Zaczynamy jednak z dwiema warstwami, tj. *wojewodztwa* i *parki*.

Aby odciążyć nieco moc obliczeniową i ustrzec się długiego czasu oczekiwania, będziemy się posługiwać centroidami obiektów z obu warstw. Zacznijmy od parków.

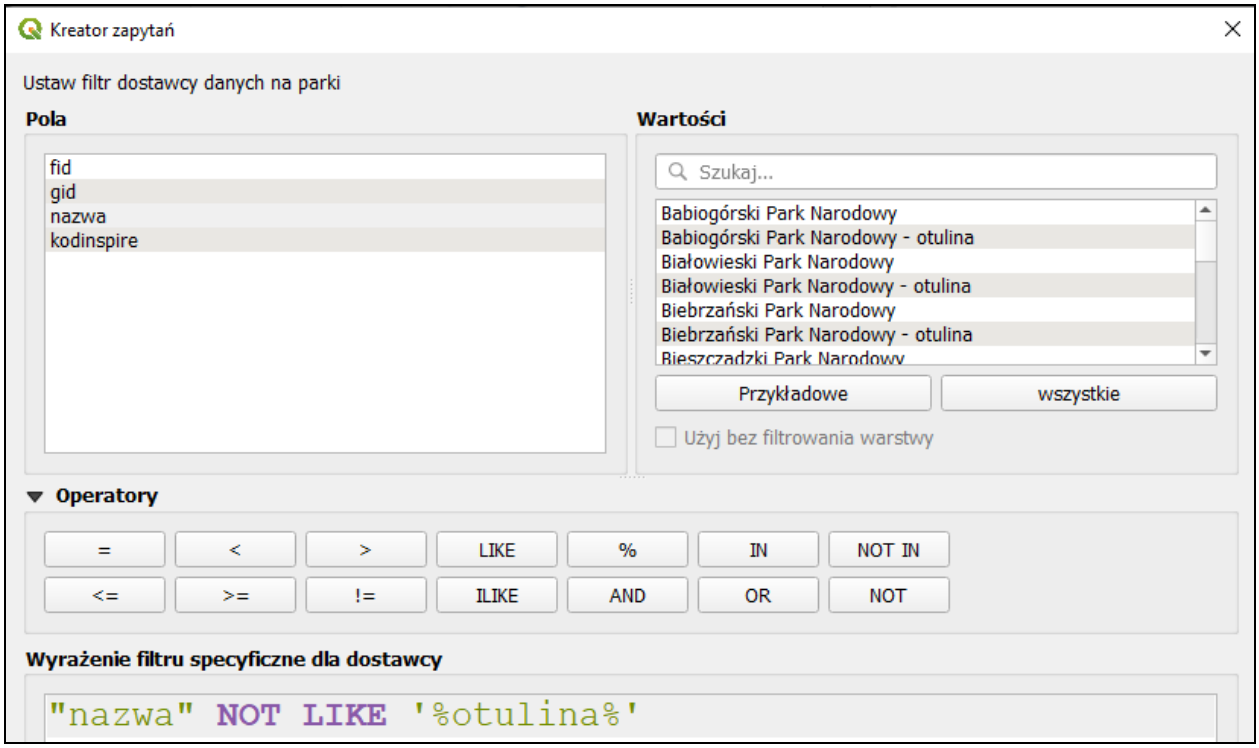
Przefiltrujmy warstwę w taki sposób, by pozbyć się geometrii reprezentujących otuliny parków. Klikamy prawym przyciskiem myszy na warstwie *parki* i z podręcznego *menu* wybieramy *Filtruj…*:



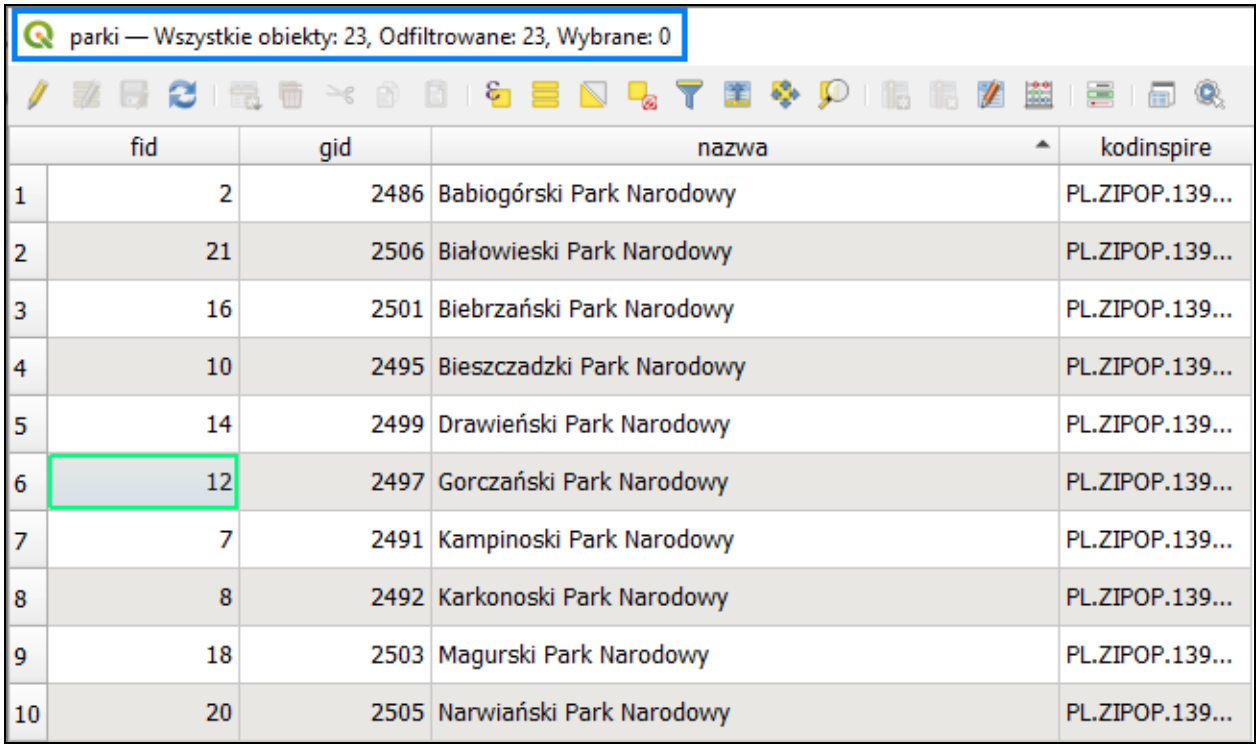
W oknie generatora filtrów przygotowujemy wyrażenie, które wykluczy obiekty zawierające w nazwie sformułowanie “otulina”. Jest to jednak część wartości, więc filtr musi być sformatowany tak, by do tej części nawiązać. Można w tym przypadku zastosować chociażby słowo kluczowe **LIKE**:



Zwróci nam to jednak rekordy, które rzeczoną cząstkę zawierają, a nam zależy na czymś zupełnie odwrotnym. Wystarczy jednak dodać słowo **NOT**, by instrukcja zwróciła oczekiwany wynik:



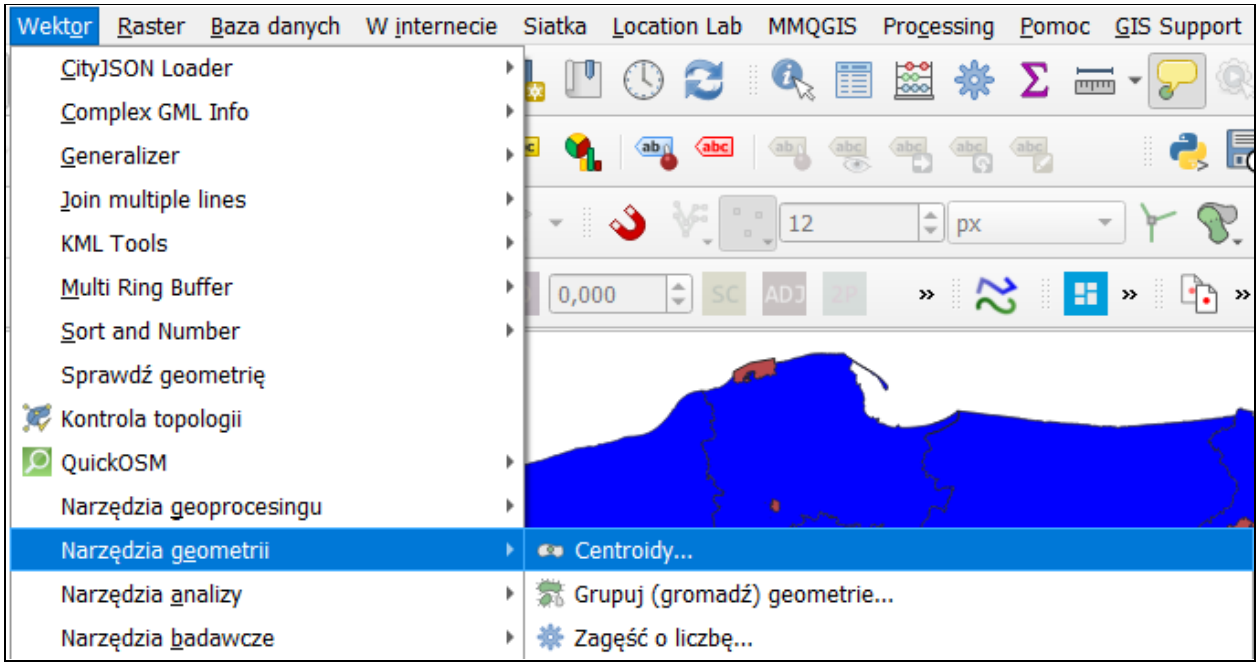
Klikamy na *OK*, zamykamy okno kreatora zapytań i przechodzimy do inspekcji tabeli atrybutów warstwy *parki*:



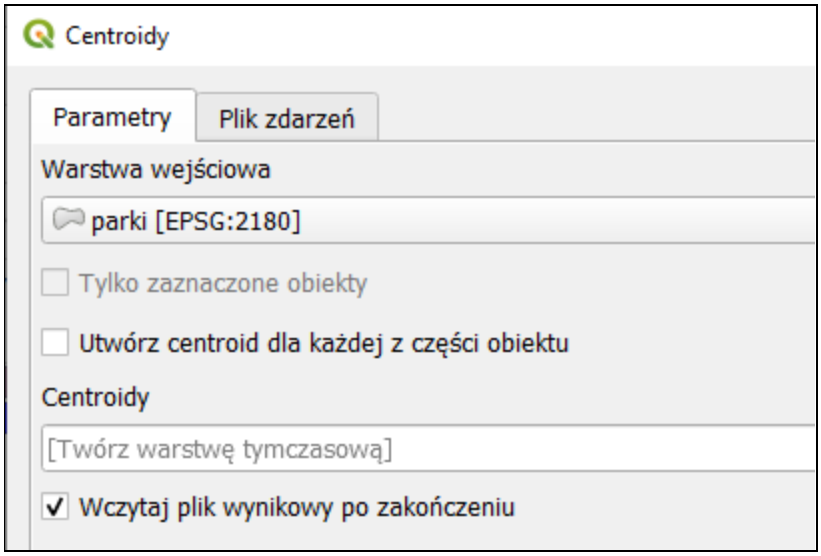
	fid	gid	nazwa	kodinspire
1	2	2486	Babiogórski Park Narodowy	PL.ZIPOP.139...
2	21	2506	Białowieski Park Narodowy	PL.ZIPOP.139...
3	16	2501	Biebrzański Park Narodowy	PL.ZIPOP.139...
4	10	2495	Bieszczadzki Park Narodowy	PL.ZIPOP.139...
5	14	2499	Drawieński Park Narodowy	PL.ZIPOP.139...
6	12	2497	Gorczański Park Narodowy	PL.ZIPOP.139...
7	7	2491	Kampinoski Park Narodowy	PL.ZIPOP.139...
8	8	2492	Karkonoski Park Narodowy	PL.ZIPOP.139...
9	18	2503	Magurski Park Narodowy	PL.ZIPOP.139...
10	20	2505	Narwiański Park Narodowy	PL.ZIPOP.139...

Po przefiltrowaniu powinna ona zawierać połowę wyjściowych danych, czyli 23 rekordy. Liczbę dostępnych wierszy możemy odczytać z paska w górnej części okna (na powyższej ilustracji zaznaczony niebieską obwódką).

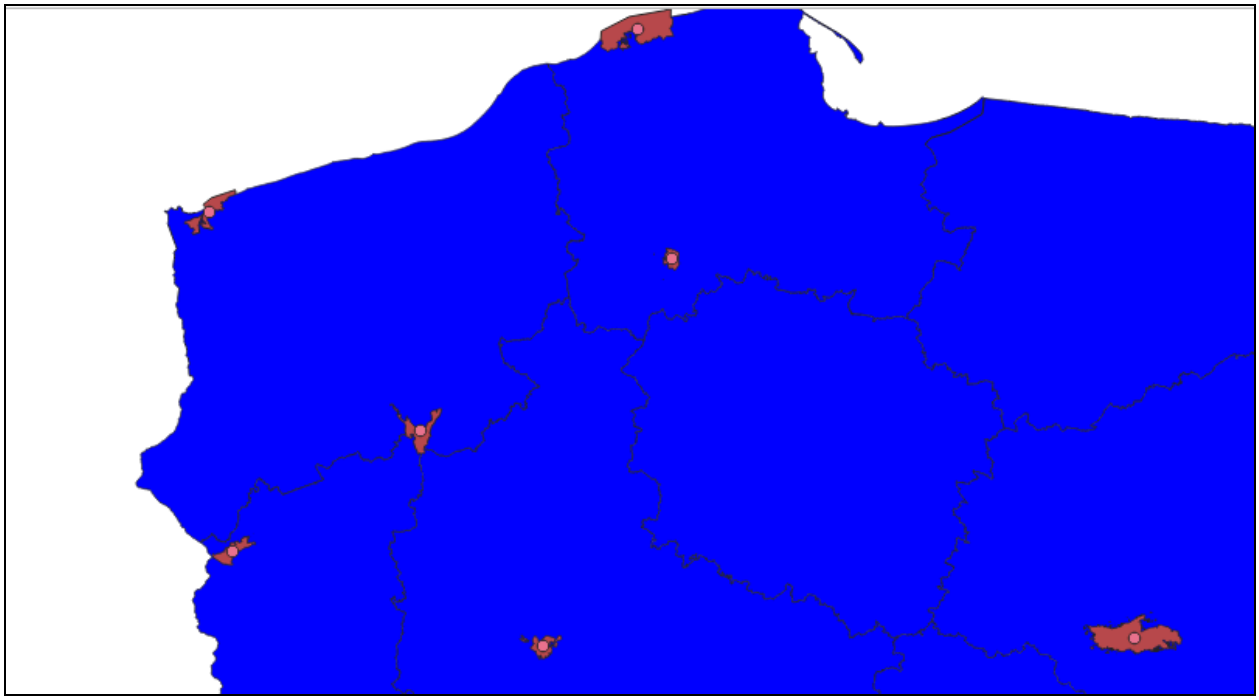
Pora na kolejny krok - tak jak wspomniano wcześniej, odciążymy nieco moc obliczeniową komputerów i wygenerujemy centroidy dla poligonów reprezentujących granice parków narodowych. W tym celu skorzystamy z narzędzia *Centroidy*. Znajduje się ono w zakładce *Wektor -> Narzędzia geometrii -> Centroidy*:



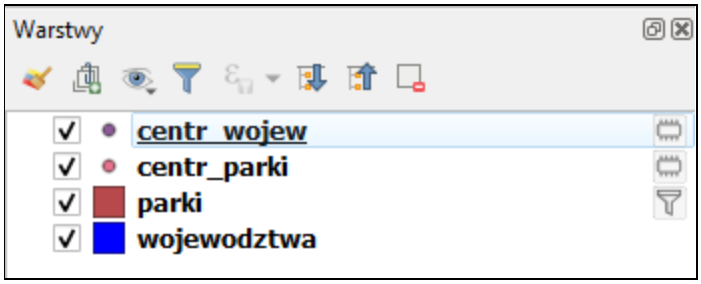
Algorytm wygeneruje nową warstwę tymczasową o geometrii punktowej. Każdy z obiektów reprezentować będzie centralny punkt poligonu z warstwy źródłowej. Narzędzie przyjmuje tylko jeden argument, tj. granice poligonów z granicami parków:



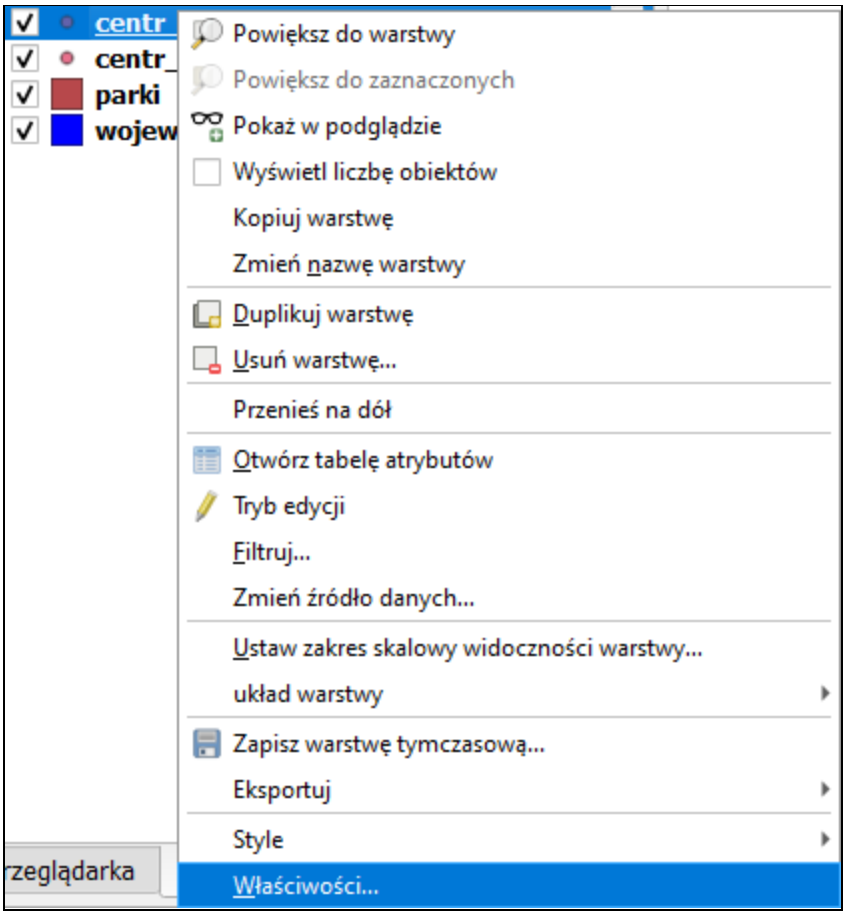
Rezultat powinien prezentować się następująco (prezentowana symbolizacja ma charakter losowy):



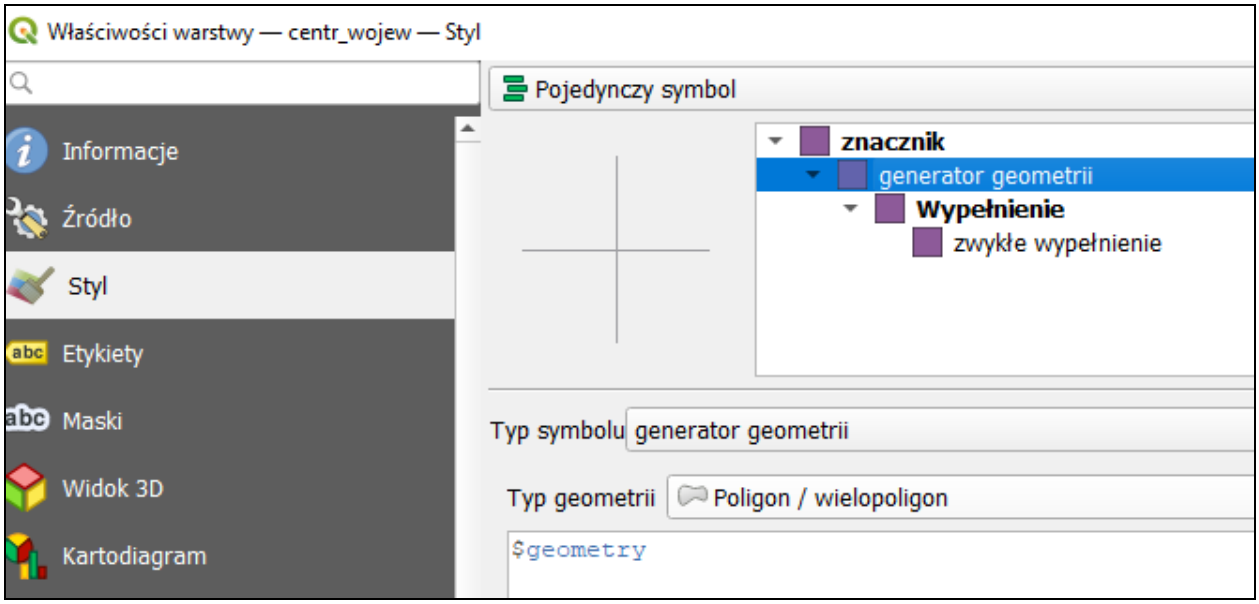
Analogiczną operację wykonujemy także na warstwie *wojewodztwa*. Powinniśmy teraz dysponować dwiema tymczasowymi warstwami punktowymi. Dla porządku zmieńmy im nazwy:



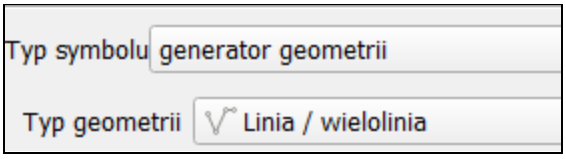
Zajmiemy się teraz symbolizacją warstwy *centr_wojew*. Klikamy prawym przyciskiem myszy na nazwie warstwy i wybieramy *Właściwości*:



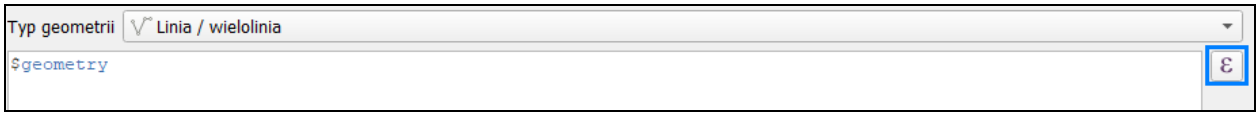
Następnie przechodzimy do zakładki *Styl* i wybieramy *generator geometrii*. Klikamy na *zwykły znacznik* i w oknie *Typ symbolu* wybieramy *generator geometrii*:



Na wstępie zmieńmy typ geometrii na linię. Pozwoli nam to śledzić na bieżąco sposób wizualizacji wyrażenia, do którego edycji przejdziemy za chwilę.



Domyślnie w oknie wyrażenia pojawia się formuła $geometry. Nasze nowe wyrażenie będzie relatywnie złożone, dlatego też będziemy je budować w oknie kreatora wyrażeń, które otwieramy klikając na:



Naszym celem jest pozyskanie listy geometrii pięciu najbliższych parków dla każdego województwa, wygenerowanie linii łączących wspomniane lokalizacje (centroidy parków i województw) oraz przekształcenie ich na obiekty poligonowe o promieniu odpowiadającym setnej części odległości (wartość ta, choć dobrana arbitralnie, pozwala na uzyskanie atrakcyjnej wizualnie geometrii końcowej). Ze względu na stopień złożoności, realizację zadania podzielimy na kilka etapów. Zacznijmy od pozyskania listy najbliższych geometrii.

Na wstępie przyjęliśmy, że bazą dla stworzenia symbolizacji będzie warstwa punktowa. Takie rozwiązanie pozwoli nam skorzystać z funkcji overlay_nearest. Zgodnie z opisem działa on dość

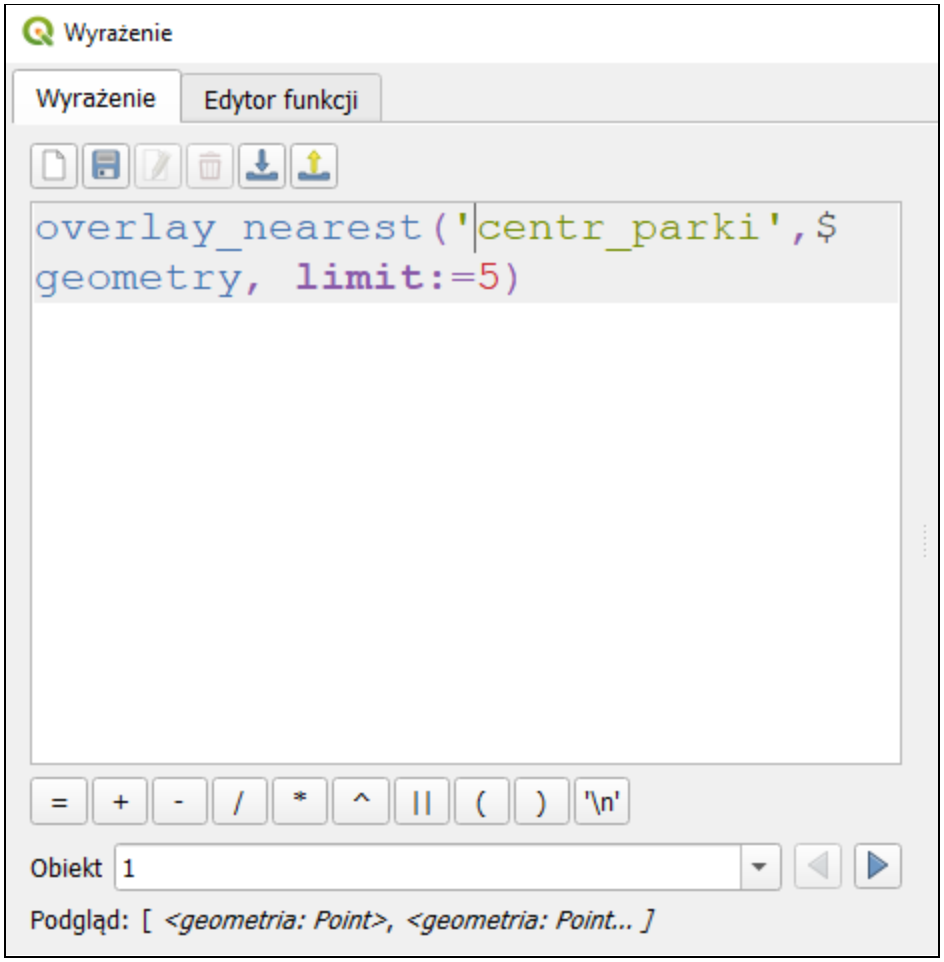
wolno, dlatego też zrezygnowaliśmy w wyjściowych geometrii poligonowych - proces analizowania relacji między nimi mógłby bowiem doprowadzić do zawieszenia programu.

Struktura funkcji przedstawia się następująco:

overlay_nearest('centr_parki',$geometry, limit:=5)

Na pierwszym miejscu wywołujemy funkcję, następnie w nawiasach jako argumenty podajemy nazwę warstwy docelowej oraz atrybut $geometry (w ten sposób adresujemy geometrie obiektów z warstwy z centroidami parków). Dodatkowo wprowadzamy limit w liczbie pięciu geometrii wynikowych (argument opcjonalny *limit:=5*).

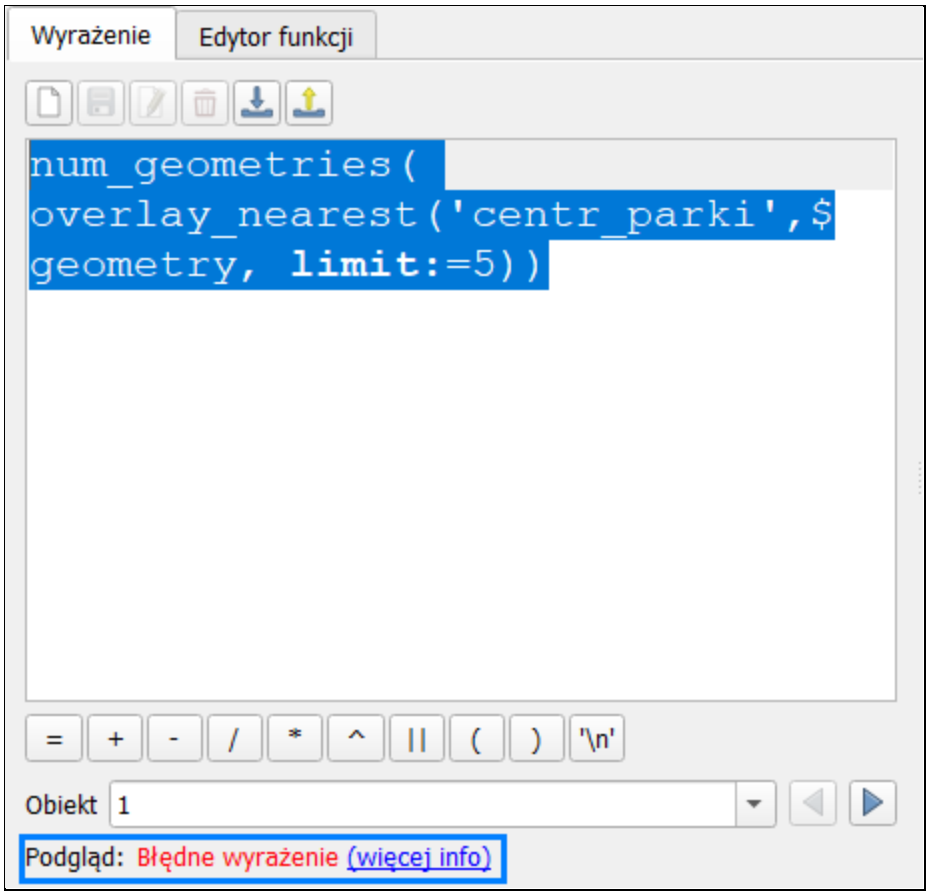
W podglądzie wyników widzimy, że wyrażenie zwraca nam listę geometrii.



Aby upewnić się, że instrukcja została wykonana poprawnie, możemy sprawdzić liczbę geometrii stosując funkcję ***num_geometries***:

num_geometries(
overlay_nearest('centr_parki',$geometry, limit:=5))

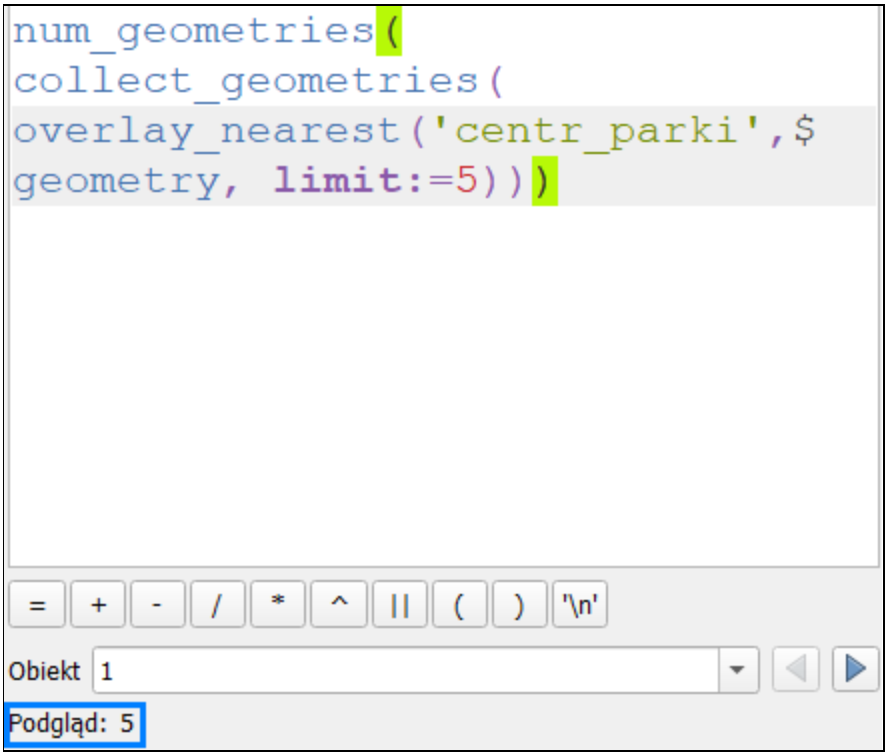
W podglądzie wyniku pojawia się jednak komunikat o błędnym wyrażeniu:



Dzieje się tak, gdyż funkcja ***num_geometries()*** przyjmuje jeden argument w postaci geometrii. Poprzednie wyrażenie zwraca wynik w formie listy, tj. ***array***. Musimy zatem znaleźć sposób na przekształcenie zbioru geometrii na geometrię wieloczęściową. Znakomicie nada się do tego funkcja ***collect()***.

collect_geometries(
overlay_nearest('centr_parki',$geometry, limit:=5))

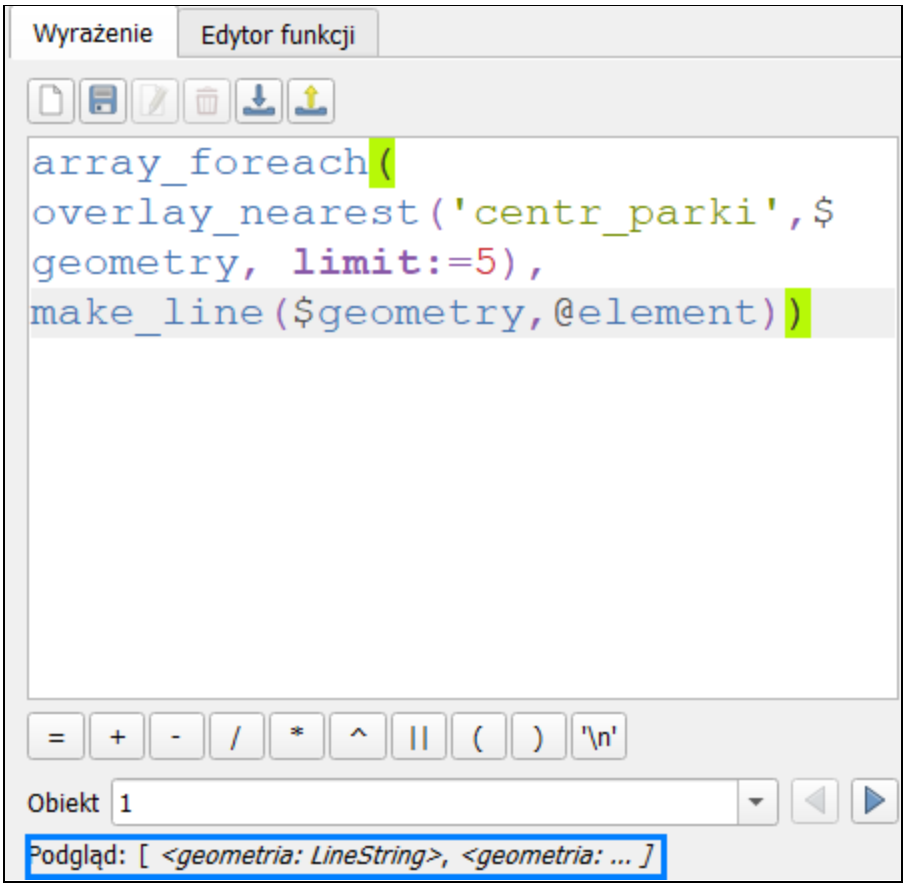
Dołóżmy do tego funkcję ***num_geometries()*** i przyjrzyjmy się podglądowi wyniku:



Liczba się zgadza, możemy więc wrócić do bazowego zapisu i przystąpić do kolejnego etapu, tj. wygenerowania linii łączącej każdy z elementów listy z punktem warstwy źródłowej. Z pomocą przychodzi nam funkcja ***array_foreach()***, która pod względem działania przypomina klasyczną pętlę - innymi słowy, wykonuje tę samą instrukcję dla każdego elementu listy. Od strony strukturalnej wygląda to następująco:

array_foreach(lista_obiektów, wyrażenie)

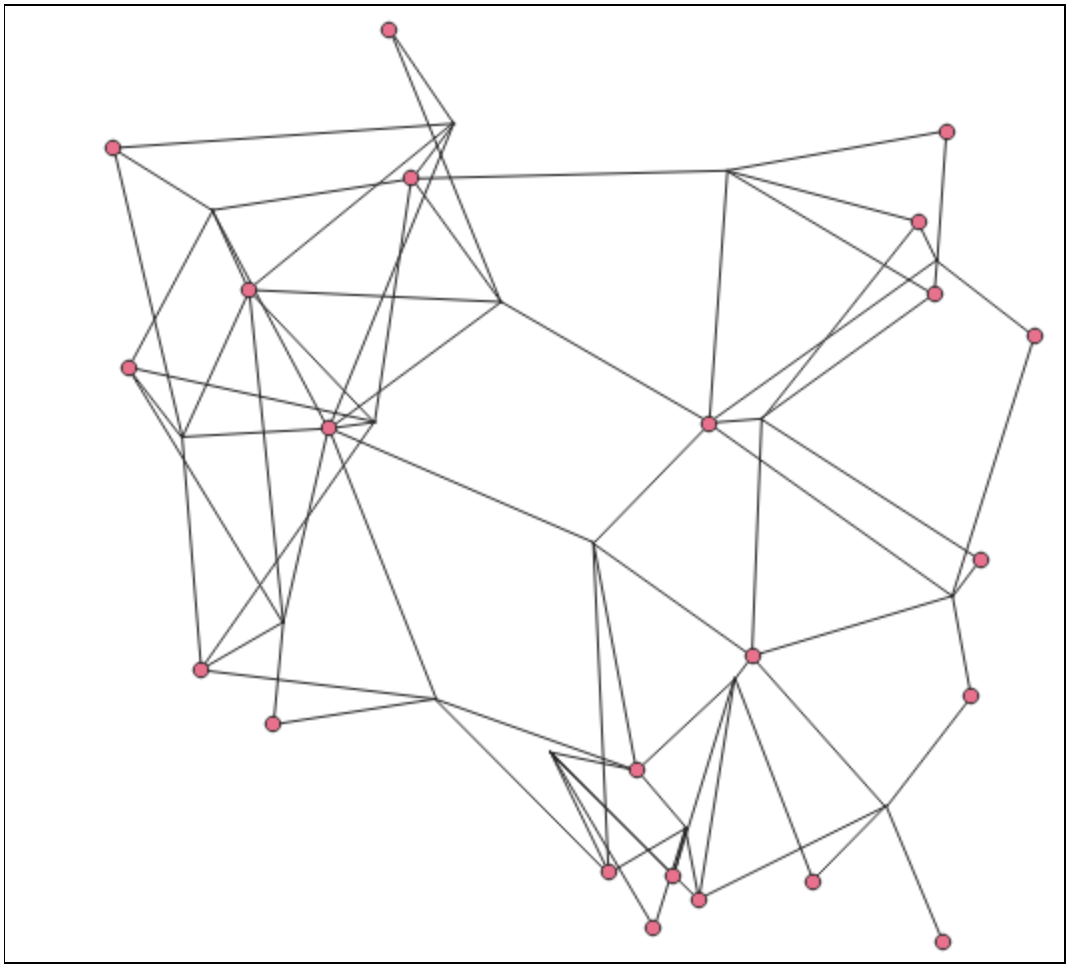
W tym przypadku listę obiektów zawiera rezultat wyrażenia ***overlay_nearest('centr_parki',$geometry, limit:=5))***. Drugim argumentem będzie funkcja ***make_line()***, tworząca geometrie liniowe między obiektami punktowymi. Pełne wyrażenie powinno wyglądać następująco:



Na tym etapie brakuje nam jeszcze funkcji, która przekształciłaby zbiór obiektów na geometrię wieloczęściową. Jest to, jak pamiętamy, ***collect_geometries()***:

collect_geometries(array_foreach(
overlay_nearest('centr_parki',$geometry, limit:=5),
make_line($geometry,@element)))

Zatwierdźmy wyrażenie klikając na *OK* i spróbujmy je zwizualizować. W tym celu w oknie edycji symbolu również klikamy na *OK* i przechodzimy do widoku mapy:



Może nie wygląda to zbyt efektownie, ale dokładnie na taki rezultat liczyliśmy. Otrzymaliśmy bowiem wizualizację linii wychodzących z jednego punktu i łączących się z pięcioma najbliższymi centroidami parków. Możemy zatem przejść do kolejnego etapu polegającego na zamianie linii na obiekty poligonowe o promieniu wartości 1/100 odległości do centroidu parku. Efekt ten osiągniemy, stosując buforowanie linii. Mamy już solidną bazę w postaci działającego kodu, możemy więc skoncentrować się na rozszerzeniu go o kolejne funkcje.

Spróbujmy teraz zmodyfikować argument *expression* funkcji ***array_foreach()***, dodając funkcję ***buffer()***:

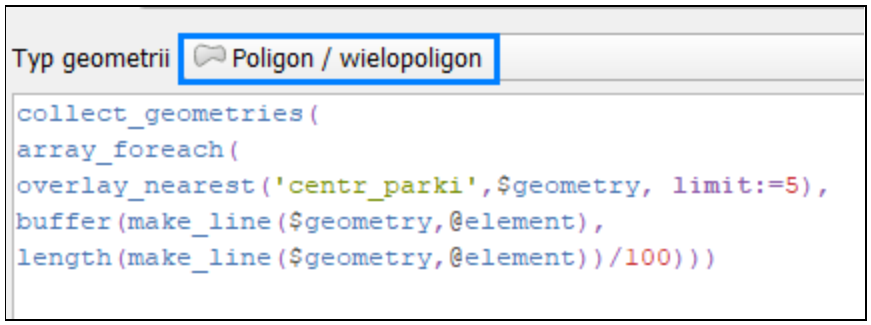
buffer(make_line($geometry,@element),
length(make_line($geometry,@element))/100))

Funkcja ta tworzy, jak sama nazwa wskazuje, bufor dla geometrii o zdefiniowanym przez użytkownika promieniu. Rolę geometrii pełni tutaj rezultat funkcji *make_line($geometry,@element)*, natomiast szerokość bufora definiuje funkcja zwracająca

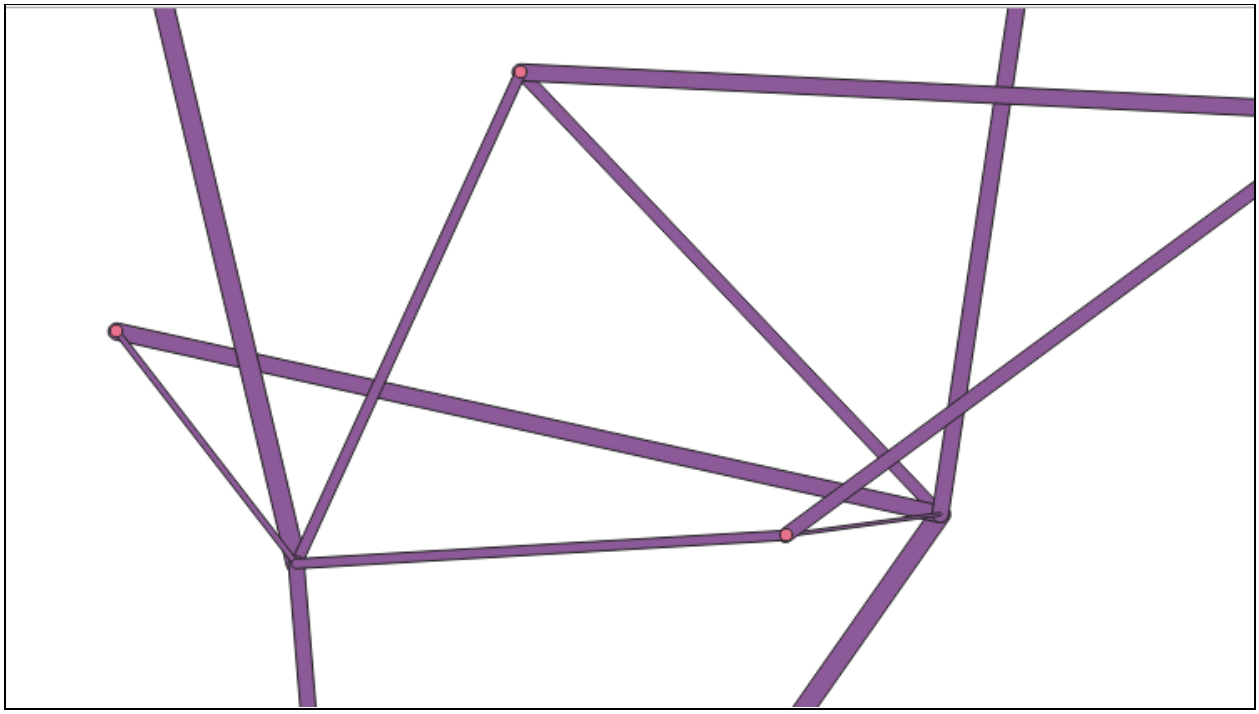
długość geometrii liniowej podzielonej przez 100:
length(make_line($geometry,@element))/100). Całe wyrażenie wygląda więc następująco:

collect_geometries(
array_foreach(
overlay_nearest('centr_parki',$geometry, limit:=5),
buffer(make_line($geometry,@element),
length(make_line($geometry,@element))/100)))

Przejdźmy teraz do jego wizualizacji. Geometria wynikowa stanowi konglomerat poligonów, musimy więc zmodyfikować typ geometrii w oknie edycji symbolu:



Tak natomiast prezentuje się sam symbol:



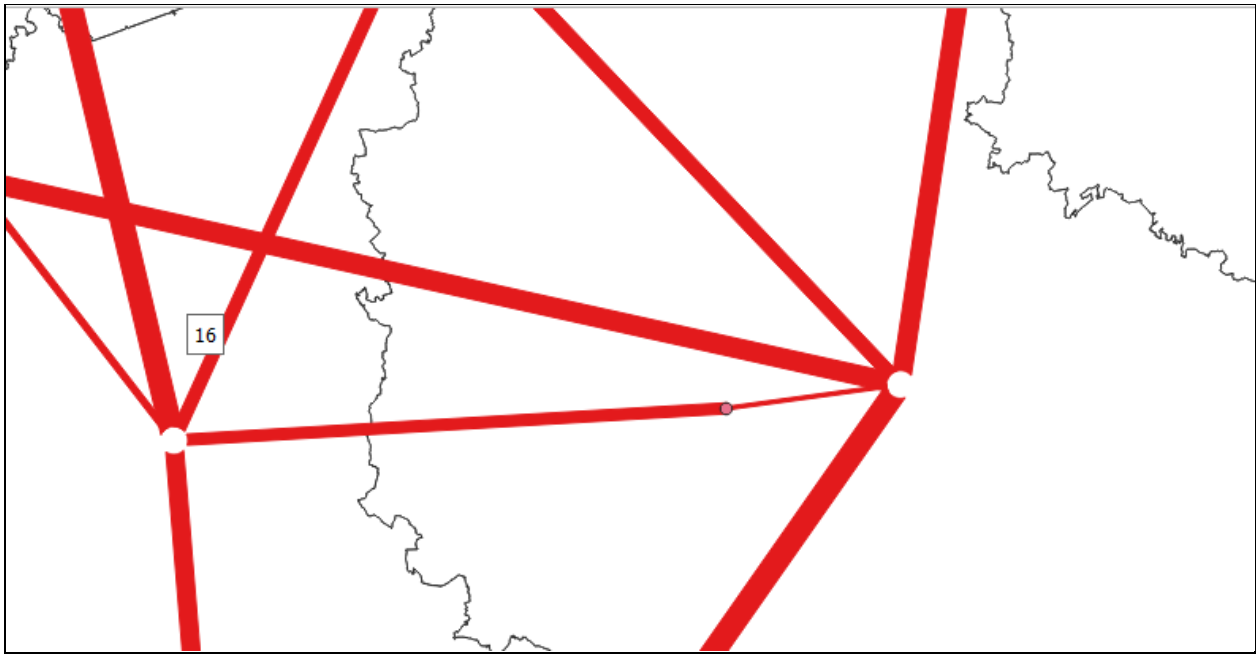
Na pierwszy rzut oka widać, że połączenia nachodzą na siebie, przez co prezentują się nieestetycznie. Spróbujmy więc dodać wokół punktów źródłowych strefę o szerokości 2500 m i zobaczmy, jak wpłynie to na symbolizację. Tym razem dokonamy modyfikacji wyrażenia bezpośrednio w oknie edycji symbolu:

Typ geometrii: Poligon / wielopoligon

```
collect_geometries(
array_foreach(
overlay_nearest('centr_parki',$geometry, limit:=5),

difference(
buffer(make_line($geometry,@element),
length(make_line($geometry,@element))/50),buffer($geometry,2500,90))))
```

Omówmy pokrótce wprowadzone modyfikacje. Po raz kolejny zmieniliśmy argument *expression*, dodając do niego funkcję ***difference()***. W efekcie pozbyliśmy się problemu nachodzących na siebie buforów. Nieestetyczne geometrie ścina bowiem kolisty bufor wygenerowany wokół punktu reprezentującego centroid województwa. Dla lepszego zwizualizowania efektu nadaliśmy mu promień rzędu 2500 m. Ponadto ujednoliciliśmy kolory wypełnienia i obrysu. Efekt działania prezentuje się następująco:

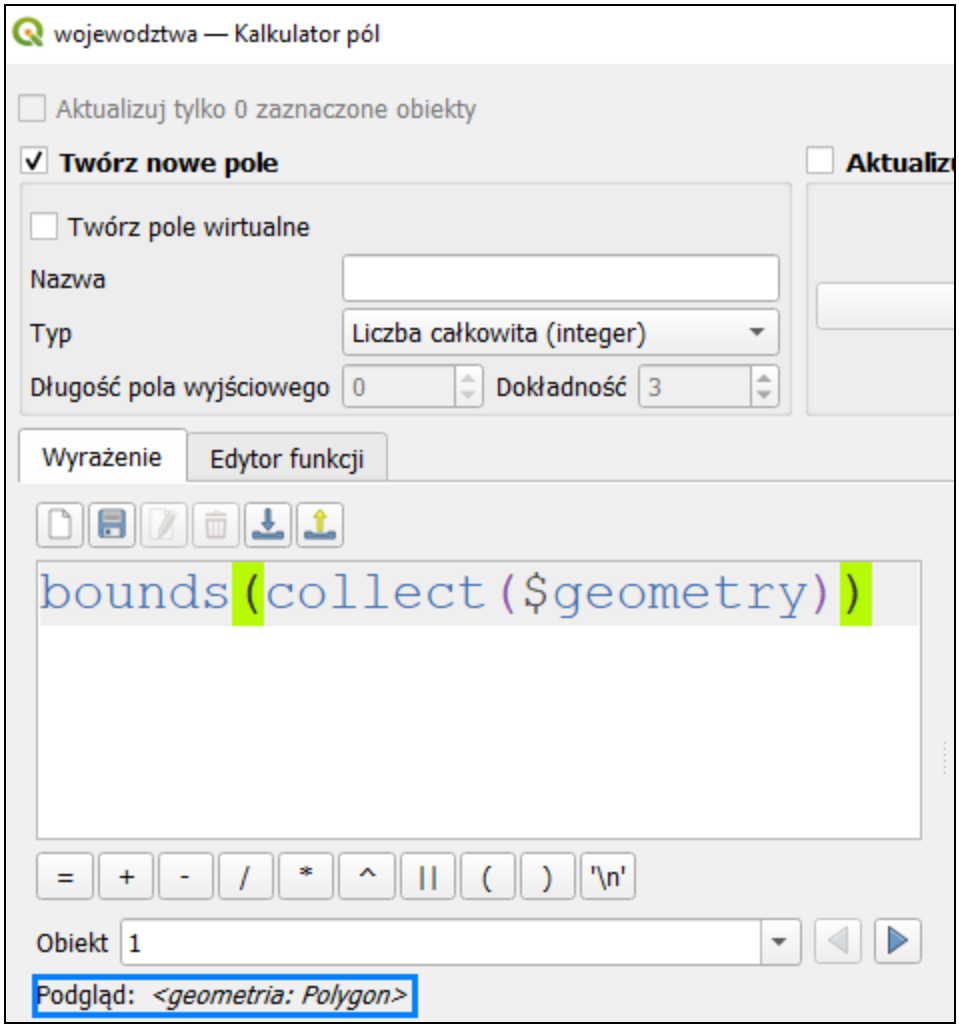


Ćwiczenie dodatkowe - przygotowanie symbolizacji z wykorzystaniem generatora geometrii

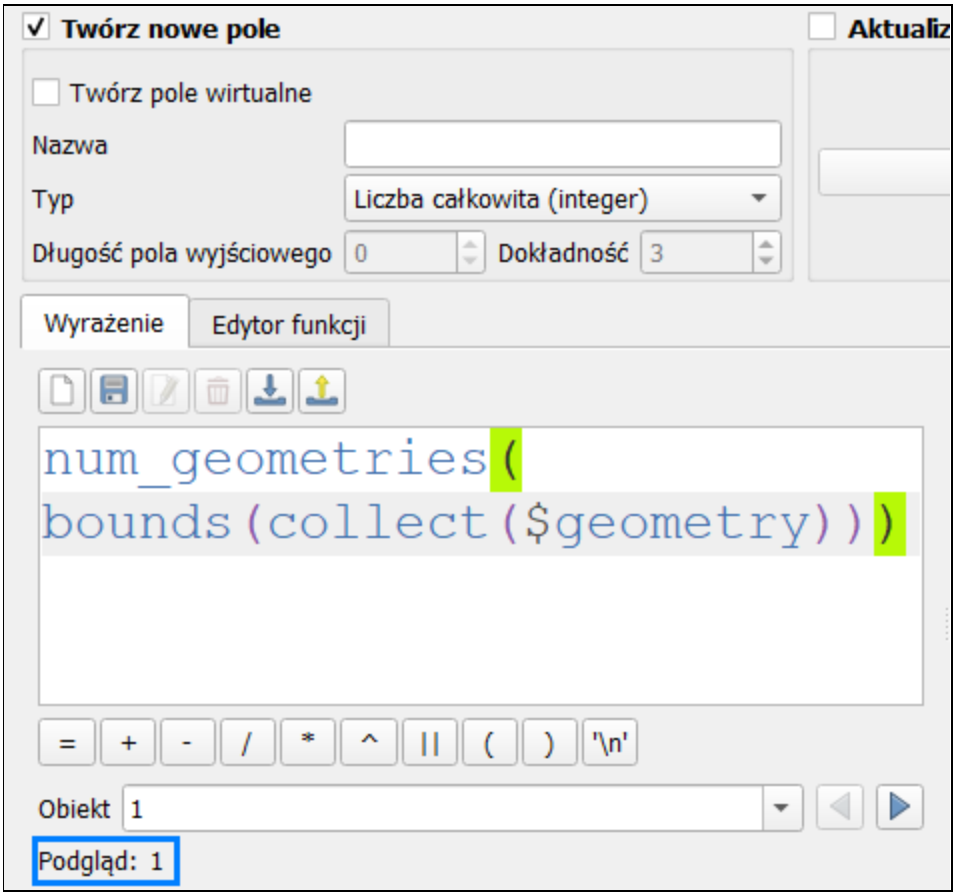
Kolejne zadanie będzie polegało na opracowaniu symbolizacji, dzięki której w sposób automatyczny zaznaczymy województwa znajdujące się w północno-zachodniej części kraju. Warunkiem włączenia do tego zbioru jest co najmniej 50% powierzchni leżącej w obrębie poligonu odpowiadającego pn-zach. ćwiartce.

Tym razem będziemy pracować na symbolizacji warstwy poligonowej *wojewodztwa*. Przechodzimy więc do właściwości, wybieramy zakładkę *Styl*, zmieniamy typ symbolu na *generator geometrii*, otwieramy okno edytora wyrażeń i przystępujemy do pracy.

Naszym pierwszym celem jest wygenerowanie prostokąta reprezentującego zasięg warstwy *wojewodztwa*. Musi on obejmować wszystkie geometrie warstwy. Efekt ten możemy osiągnąć, stosując funkcję ***collect()***. Do wydobycia zasięgu w postaci prostokąta otaczającego użyjemy natomiast funkcji ***bounds()***:



Podgląd wyniku wygląda obiecująco. Sprawdźmy jednak, czy rezultat zawiera wyłącznie jedną geometrię. Do tego celu wykorzystamy funkcję ***num_geometries()***:

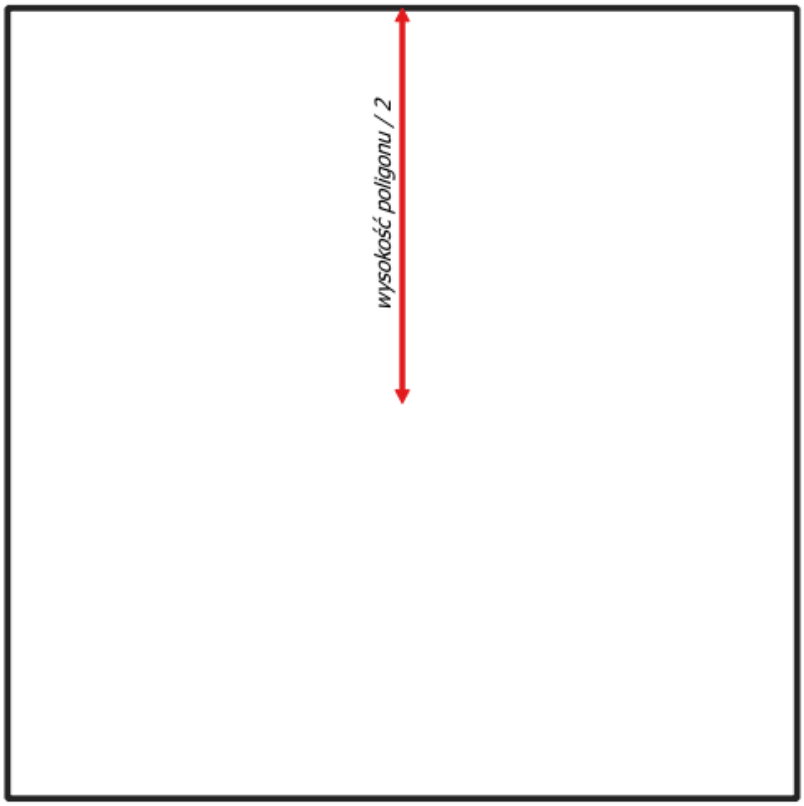


W podglądzie wyświetla się wartość 1. Instrukcja zwraca więc poprawną liczbę geometrii.

Kolejny krok to wygenerowanie centroidu dla utworzonego zasięgu. Dysponując jego geometrią będziemy mogli wyznaczyć trzy pozostałe wierzchołki prostokąta reprezentującego północno-zachodnią ćwiartkę:

centroid(
bounds(collect($geometry)))

Spróbujmy napisać teraz roboczy fragment kodu, który wygeneruje linię łączącą centroid poligonu reprezentującego pełen zasięg warstwy z prawym górnym wierzchołkiem ćwiartki. Podstawą będzie funkcja ***make_line()***. Wymaga ona jednak co najmniej dwóch argumentów, czyli punktów, które zostaną połączone linią prostą. W jaki sposób wygenerować geometrię wspomnianego wierzchołka? Wiemy, że musi być on przesunięty na osi pionowej o odległość równą połowie wysokości prostokąta. Przedstawia to poniższa ilustracja:



Przejdźmy więc do kodu. Zaczynamy od wygenerowania centroidu, który będzie jednocześnie pierwszym wierzchołkiem projektowanej linii:

centroid(bounds(collect($geometry)))

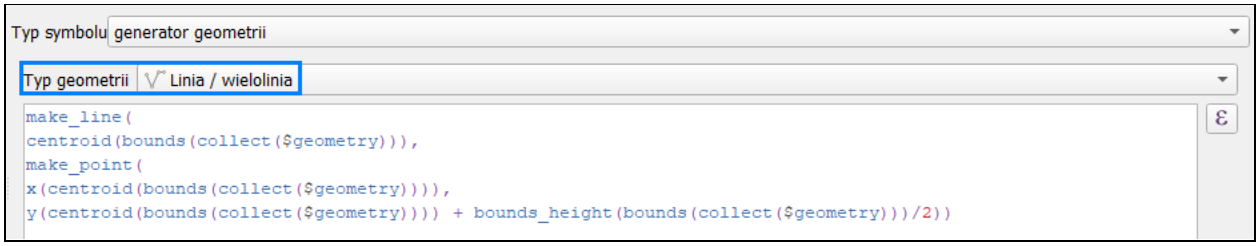
Na końcu bieżącej formuły wstawiamy przecinek i przechodzimy do wygenerowania drugiego wierzchołka. Posłużymy się funkcją ***make_point()***, przyjmującą dwa argumenty w postaci koordynatów x i y projektowanej geometrii punktowej. Jak łatwo zauważyć, współrzędna x nie ulegnie zmianie, natomiast współrzędna y powinna być przesunięta w stosunku do źródłowej o wartość odpowiadającą połowie wysokości prostokąta reprezentującego zasięg. Operację tę możemy zapisać w następujący sposób:

make_point(
x(centroid(bounds(collect($geometry)))),
y(centroid(bounds(collect($geometry)))) + bounds_height(bounds(collect($geometry)))/2)

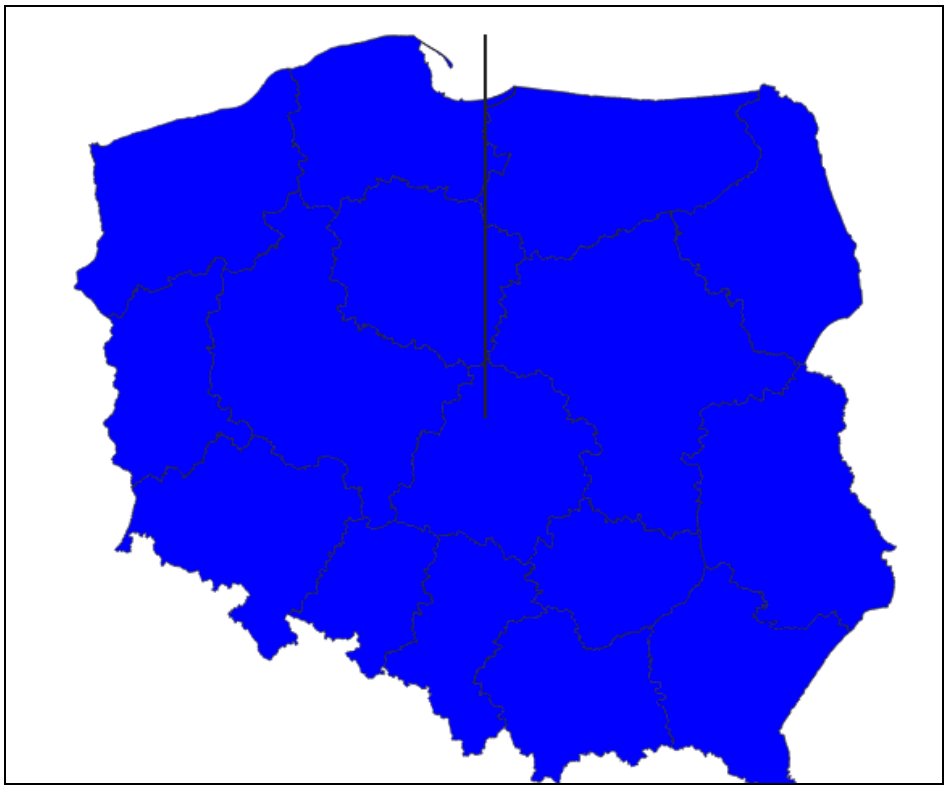
gdzie ***bounds_height()*** to funkcja, która zwraca wysokość poligonu. Połączmy teraz poszczególne elementy w całość i sprawdźmy, czy uda nam się poprawnie wygenerować linię:

make_line(
centroid(bounds(collect($geometry))),
make_point(
x(centroid(bounds(collect($geometry)))),
y(centroid(bounds(collect($geometry)))) + bounds_height(bounds(collect($geometry)))/2))

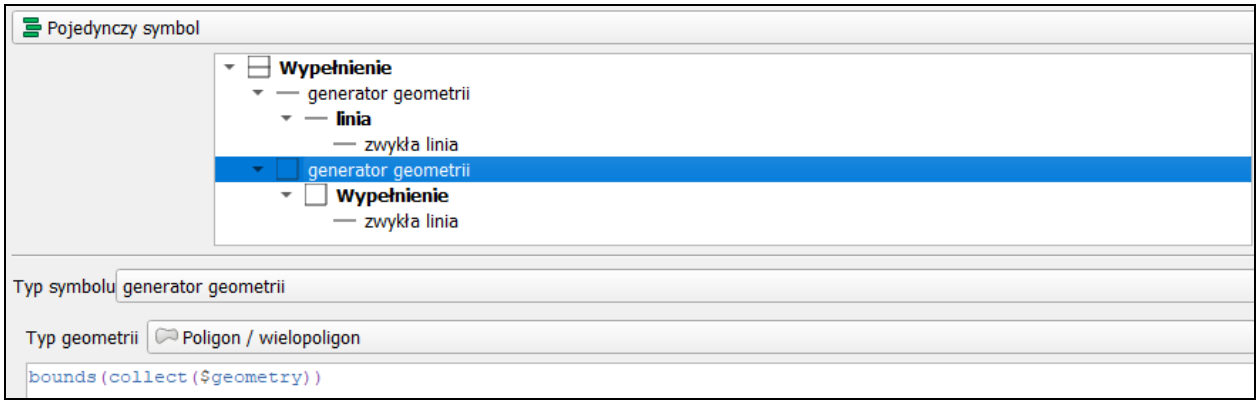
Zatwierdzamy formułę i dopasowujemy typ geometrii symbolu:



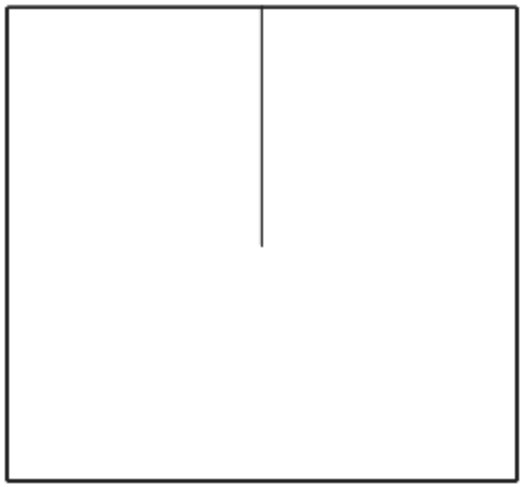
Możemy teraz zatwierdzić zmiany i wrócić do widoku mapy. Powinniśmy uzyskać widok zbliżony do poniższego:



Obraz wynikowy stanie się bardziej czytelny, jeśli zwykłe wypełnienie zamienimy na generator geometrii w typie poligonu z następującą formułą:



Ponownie zatwierdźmy zmiany i jeszcze raz skontrolujmy symbolizację warstwy w głównym oknie mapy:



Może nie wygląda efektownie, ale dokładnie o taki rezultat nam chodziło. Widzimy bowiem wyraźnie, że wygenerowana linia wyznacza prawą krawędź północno-zachodniej ćwiartki.

Skoro potrafimy odpowiednio opisać współrzędne wierzchołków interesującej nas części, możemy przystąpić do wygenerowania jej pełnego zasięgu. Wykorzystamy do tego funkcję ***make_rectangle_3points()***. Przyjmuje ona trzy argumenty w postaci współrzędnych wierzchołków projektowanego prostokąta. Kolejność podawania koordynatów ma znaczenie - zaczynamy od lewego górnego punktu, kończymy zaś na centroidzie. Aby uniknąć kopiowania powtarzających się fragmentów kodu, wprowadzimy do naszej formuły nową funkcję ***with_variable().*** Umożliwia ona utworzenie lokalnej zmiennej o zdefiniowanej przez użytkownika nazwie, a następnie wykorzystanie jej w wyrażeniu stanowiącym trzeci argument funkcji. Brzmi to skomplikowanie, zacznijmy więc od prostego przykładu.

with_variable('suma',10 + 35,@suma + 11)

Zastosowaliśmy funkcję ***with_variable()*** do utworzenia zmiennej *suma*, której przypisaliśmy wartość w postaci sumy dwóch liczb całkowitych; w trzecim argumencie funkcji wykonujemy wyrażenie, w którym przywołujemy wcześniej utworzoną zmienną stosując znak @ wraz z jej nazwą. W rezultacie otrzymujemy wynik 56, będący sumą wartości @suma oraz liczby 11. Omawiana funkcja znajduje zastosowanie w sytuacji, gdy w jednym wyrażeniu musimy wielokrotnie użyć wartości reprezentowanej przez inne wyrażenie o skomplikowanej strukturze. Dzięki zastosowaniu zmiennej mamy dużo większą kontrolę nad poprawnością zapisu (np. nad liczbą nawiasów).

Jak jednak mielibyśmy wykorzystać tę funkcję w naszym ćwiczeniu? Możemy na przykład uprościć kod wprowadzając zmienną @*środek*, której przypiszemy wartość w postaci geometrii reprezentującej centroid pełnego zasięgu warstwy. Następnie wykorzystamy ją do wygenerowania północno-zachodniej ćwiartki. Pełny kod prezentuje się następująco:

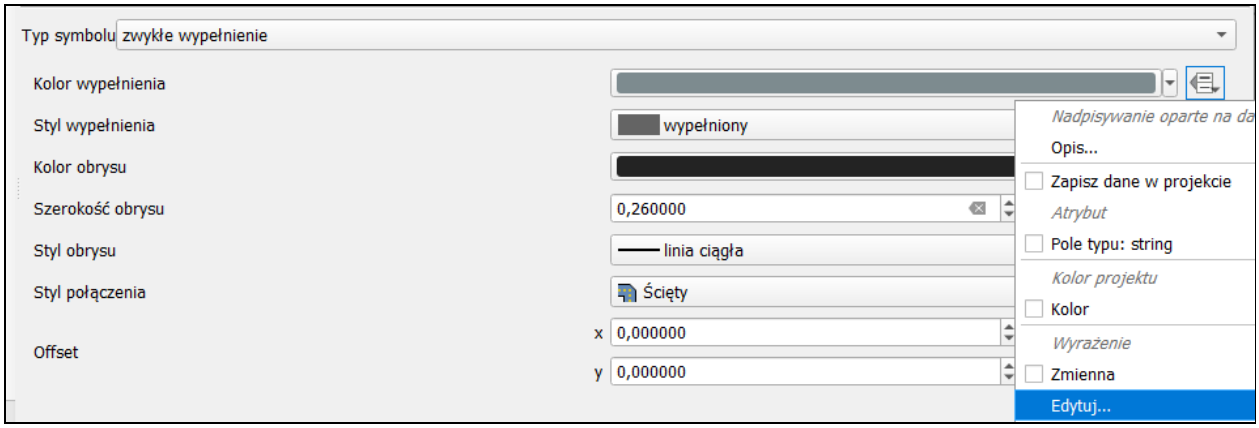
```
with_variable(
'środek',
centroid(bounds(collect($geometry))),
make_rectangle_3points(

make_point(
x(@środek) - bounds_width(bounds(collect($geometry)))/2,
y(@środek) + bounds_height(bounds(collect($geometry)))/2),
make_point(
x(@środek),
y(@środek) + bounds_height(bounds(collect($geometry)))/2),

@środek))
```

Wygenerowaliśmy trzy wierzchołki, które następnie podaliśmy jako argumenty funkcji ***make_rectange_3points()***. Koordynaty wierzchołków obliczyliśmy wykorzystując dane o wysokości (bounds_height) oraz szerokości (bounds_width) zasięgu warstwy z województwami. Do osiągnięcia rezultatu końcowego brakuje nam jeszcze funkcji, która zaznaczy województwa leżące w wygenerowanej ćwiartce. Pamiętajmy przy tym, że zasięg województwa musi pokrywać się z zasięgiem ćwiartki w więcej niż 50 procentach. Skopiujmy naszą formułę i zamknijmy teraz okno generatora geometrii. Przypiszmy naszej warstwie styl domyślny, tj. pojedynczy symbol wraz losowym wypełnieniem.

Jednocześnie przy polu z kolorem wypełnienia rozwińmy opcję *Nadpisywanie oparte na danych* i wybierzmy *Edytuj*:



Następnie wprowadźmy do formuły funkcję ***intersection()*** w celu obliczenia wspólnych części geometrii obu warstw:

```
intersection($geometry,
with_variable(
'środek',
centroid(bounds(collect($geometry))),
make_rectangle_3points(
make_point(
x(@środek) - bounds_width(bounds(collect($geometry)))/2,
y(@środek) + bounds_height(bounds(collect($geometry)))/2),
make_point(
x(@środek),
y(@środek) + bounds_height(bounds(collect($geometry)))/2),

@środek)))
```

Dodajmy jeszcze funkcję ***area()*** i zaznaczmy, że stosunek powierzchni przecięcia do powierzchni województwa musi być większy niż 50%:

```
area(
intersection($geometry,
with_variable(
'środek',
centroid(bounds(collect($geometry))),
make_rectangle_3points(
make_point(
x(@środek) - bounds_width(bounds(collect($geometry)))/2,
y(@środek) + bounds_height(bounds(collect($geometry)))/2),
make_point(
```

x(@środek),
y(@środek) + bounds_height(bounds(collect($geometry)))/2),

@środek)))) / $area * 100 > 50

Tak sformułowany warunek możemy wykorzystać jako argument funkcji ***if()***. Przyjmijmy, że województwa z ćwiartki mają mieć kolor pomarańczowy, pozostałe natomiast - szary:

if(area(
intersection($geometry,
with_variable(
'środek',
centroid(bounds(collect($geometry))),
make_rectangle_3points(
make_point(
x(@środek) - bounds_width(bounds(collect($geometry)))/2,
y(@środek) + bounds_height(bounds(collect($geometry)))/2),
make_point(
x(@środek),
y(@środek) + bounds_height(bounds(collect($geometry)))/2),

@środek)))) / $area * 100 > 50,'orange','grey')

Klikamy na *OK* i przechodzimy do kontroli poprawności naszej wizualizacji:

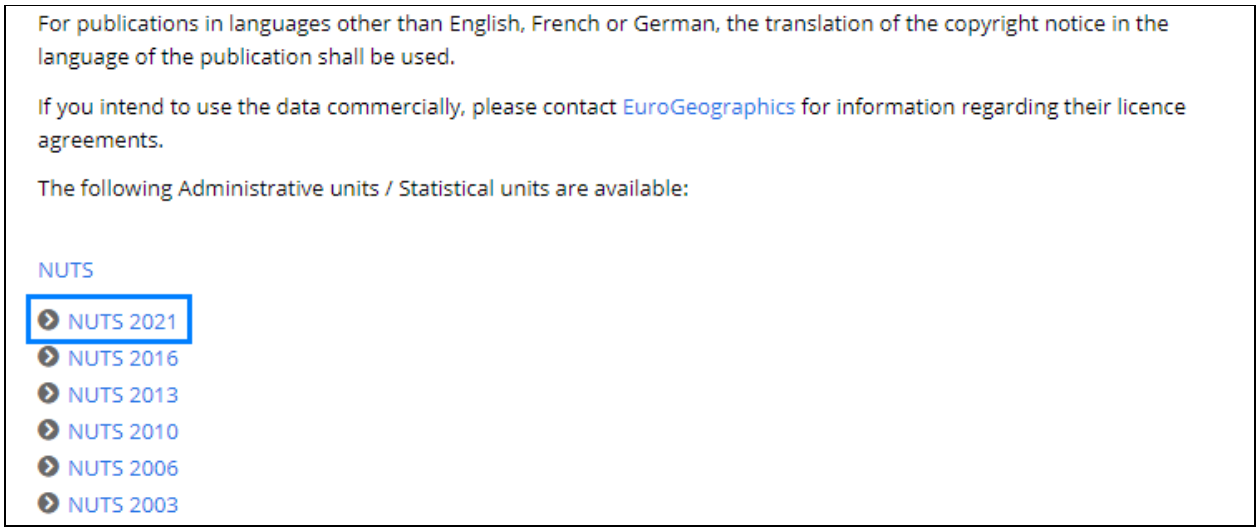
Podstawy web mappingu w QGIS. Praktyczne wykorzystanie wtyczki QGIS2Web

Celem ćwiczenia jest stworzenie prostej mapy internetowej, działającej w oknie przeglądarki, zawierającej wybrany podkład mapowy oraz warstwę wektorową o zdefiniowanej przez użytkownika symbolizacji. Projekt mapy zostanie przygotowany w programie QGIS, a następnie wyeksportowany do formatu .html przy użyciu wtyczki ***qgis2web***. Realizacja ćwiczenia przebiegać będzie etapowo:

1. Utworzenie nowego projektu oraz dodanie danych
2. Edycja pól w *Tabeli Atrybutów*, przygotowanie symbolizacji
3. Instalacja wtyczki qgis2web; edycja ustawień wtyczki; eksport do formatu wynikowego.

Etap pierwszy

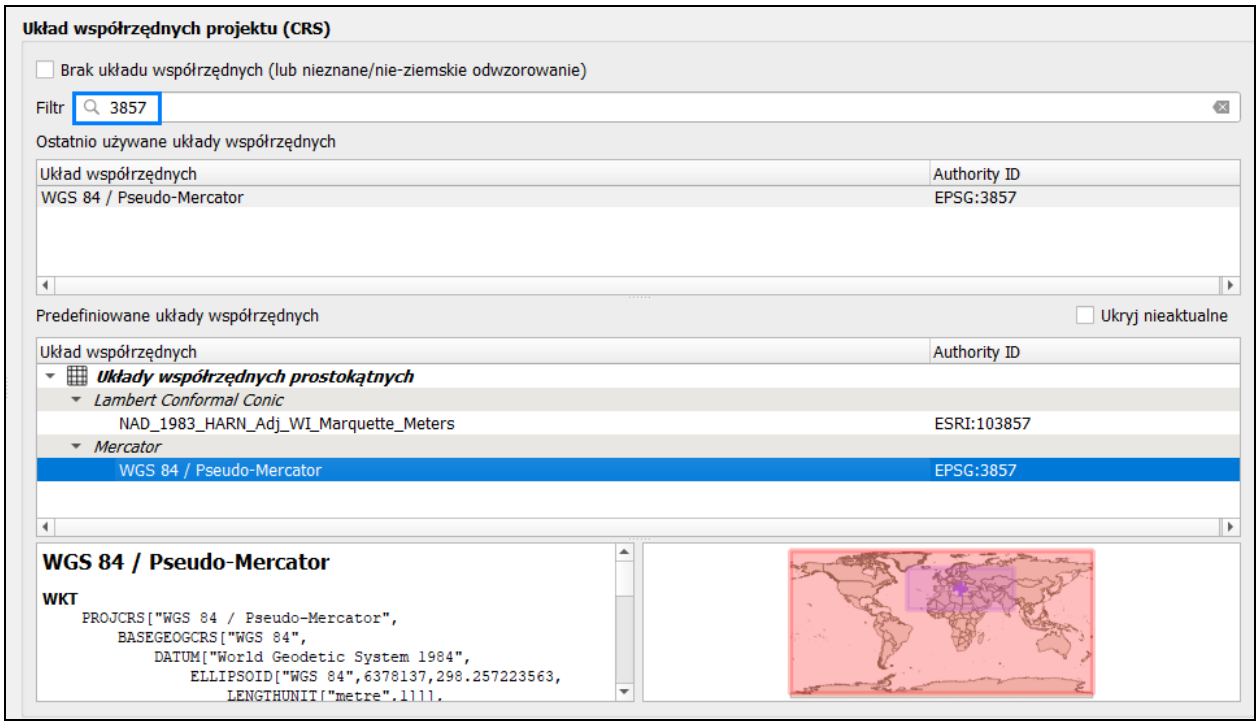
Naszym zadaniem jest przygotowanie mapy przedstawiającej lokalizacje wulkanów znajdujących w granicach Unii Europejskiej. Plik z wulkanami, zapisany w formacie ESRI Shapefile, znajduje się w katalogu z danymi, natomiast granice administracyjne państw unijnych można pobrać ze strony https://ec.europa.eu/eurostat/web/gisco/geodata/reference-data/administrative-units-statistical-units . Aby pobrać warstwy, należy kliknąć na:



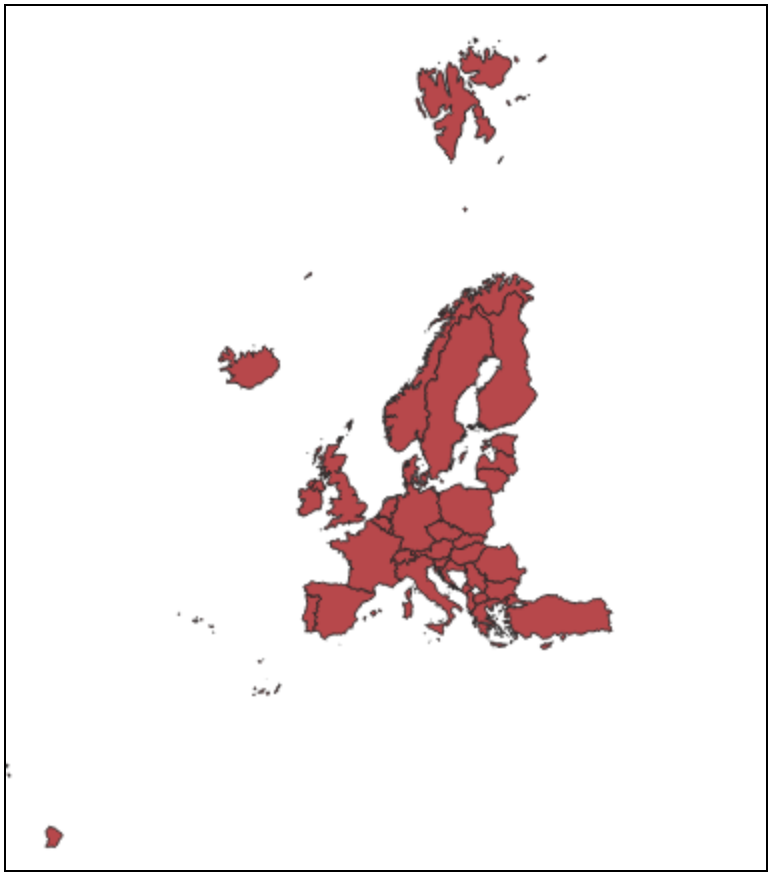
Następnie wybieramy odpowiednią pozycję z listy (pobieramy dane w formacie .shp i skali 1:10m):

Administrative or Statistical unit	Version date	Scale	File format to download					API
			SHP	TopoJSON	geoJSON	GDB	SVG	
NUTS 2021	01/02/2020	1:1 Million	ZIP	ZIP	ZIP	ZIP	ZIP	
		1:3 Million	ZIP	ZIP	ZIP	ZIP	ZIP	
		1:10 Million	ZIP	ZIP	ZIP	ZIP	ZIP	
		1:20 Million	ZIP	ZIP	ZIP	ZIP	ZIP	
		1:60 Million	ZIP	ZIP	ZIP	ZIP	ZIP	

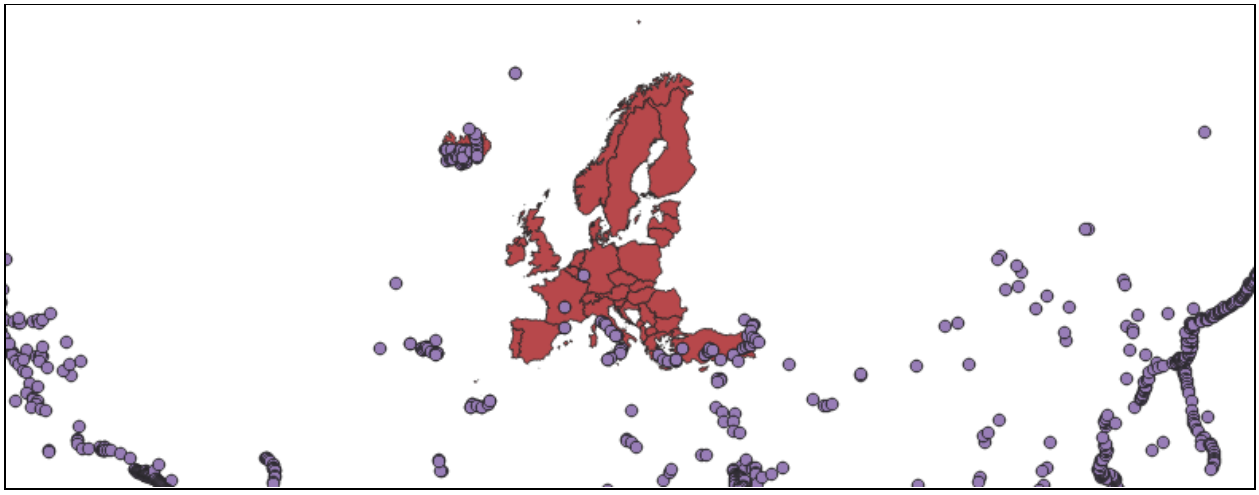
Po rozpakowaniu danych należy dodatkowo wypakować pliki z archiwum o nazwie NUTS_RG_10M_2021_3857_LEVL_0.shp. Następnie uruchamiamy program QGIS w wersji *desktop*. Tworzymy nowy projekt w układzie współrzędnych EPSG:3857:



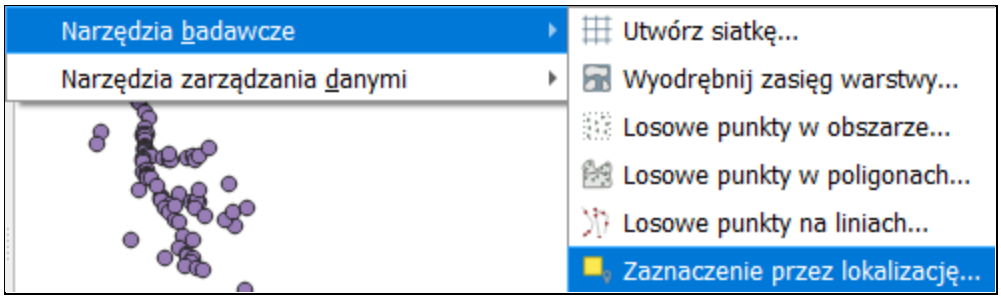
Przechodzimy do dodania granic administracyjnych. Do okna mapy przeciągamy wcześniej rozpakowany plik NUTS_RG_10M_2021_3857_LEVL_0.shp. Widok projektu po wykonaniu działania powinien przedstawiać się następująco (naturalnie uwaga ta nie dotyczy losowo dobranej palety barw) :



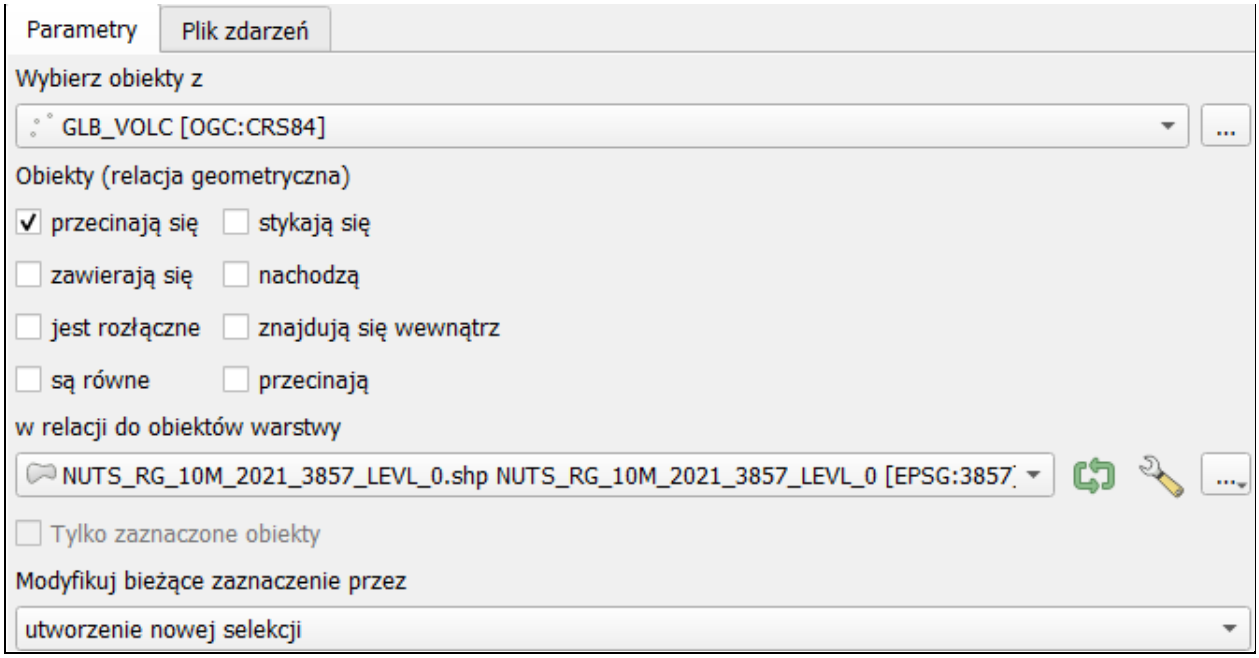
Granice administracyjne posłużą nam do selekcji, a następnie wyeksportowania tylko tych wulkanów, które leżą w obrębie terytoriów państw członkowskich. Dodajemy do projektu warstwę GLB_VOLC.shp - znajduje się ona w przekazanym katalogu z danymi.



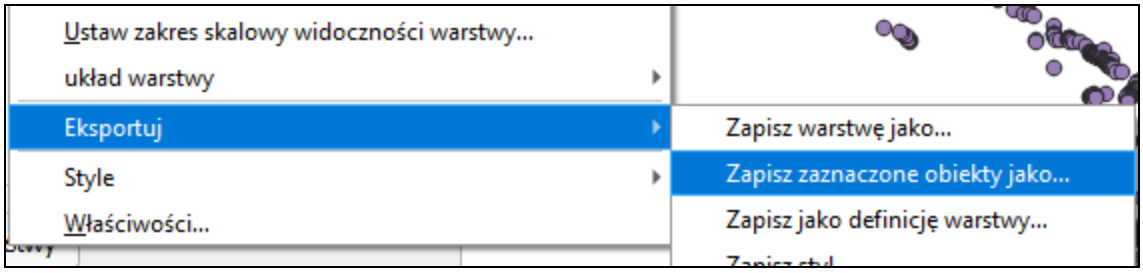
Z zakładki *Wektor -> Narzędzia badawcze* wybieramy narzędzie *Zaznaczenie przez lokalizację*:



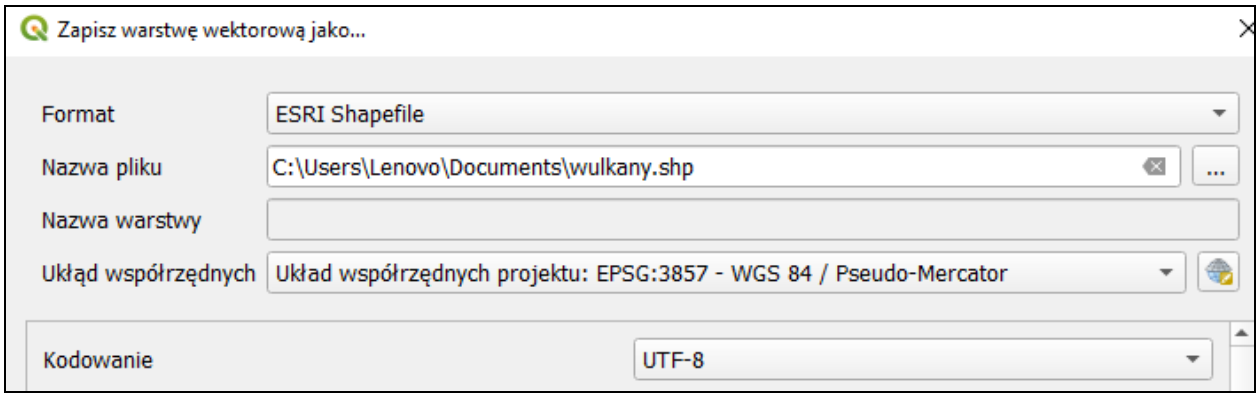
W oknie algorytmu jako warstwę wejściową wskazujemy GLB_VOLC, natomiast jako warstwę nakładki NUTS_RG(...). Interesują nas wszystkie obiekty z pierwszej warstwy, których geometrie przecinają się z geometriami drugiej, dlatego też spośród relacji geometrycznych wybieramy opcję *przecinają się*:



Efektem końcowym działania jest utworzenie nowej selekcji w obrębie warstwy z wulkanami. Zaznaczone obiekty należy wyeksportować do nowej warstwy w formacie .shp, której nadamy nazwę wulkany:



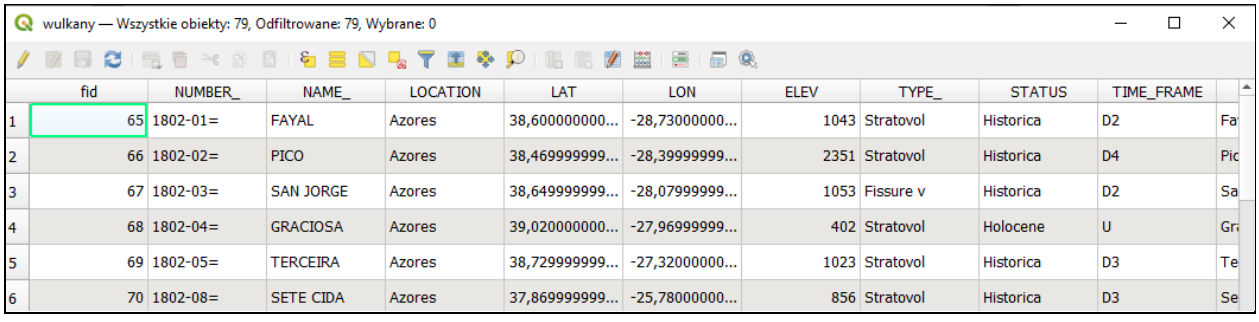
Przed eksportem należy dodatkowo ustawić układ współrzędnych warstwy wynikowej. Wybieramy ten o kodzie EPSG:3857:



Po zapisaniu i dodaniu do projektu nowoutworzonej warstwy (działanie to odbywa się automatycznie) możemy usunąć warstwy GLB_VOLC i NUTS_RG(...).

Etap drugi

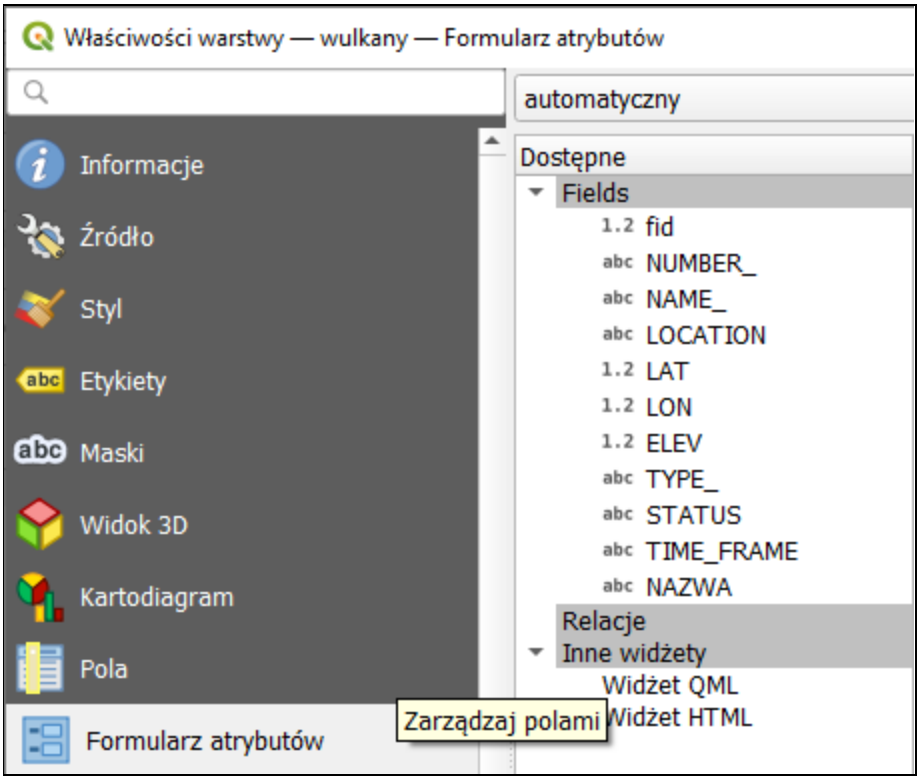
Otwórzmy teraz tabelę atrybutów warstwy z wulkanami (europejskimi):



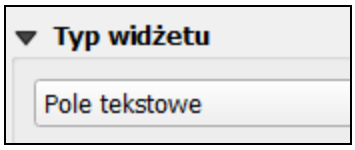
	fid	NUMBER_	NAME_	LOCATION	LAT	LON	ELEV	TYPE_	STATUS	TIME_FRAME	
1	65	1802-01=	FAYAL	Azores	38,600000000...	-28,73000000...	1043	Stratovol	Historica	D2	Fa
2	66	1802-02=	PICO	Azores	38,469999999...	-28,39999999...	2351	Stratovol	Historica	D4	Pic
3	67	1802-03=	SAN JORGE	Azores	38,649999999...	-28,07999999...	1053	Fissure v	Historica	D2	Sa
4	68	1802-04=	GRACIOSA	Azores	39,020000000...	-27,96999999...	402	Stratovol	Holocene	U	Gr
5	69	1802-05=	TERCEIRA	Azores	38,729999999...	-27,32000000...	1023	Stratovol	Historica	D3	Te
6	70	1802-08=	SETE CIDA	Azores	37,869999999...	-25,78000000...	856	Stratovol	Historica	D3	Se

Nie wszystkie kolumny zawierają istotne dla nas informacje. Nie musimy jednak usuwać mniej istotnych atrybutów - warto zaznaczyć, że taka operacja stanowi ingerencję w strukturę warstwy, co nie zawsze jest działaniem pożądanym. W tym przypadku skorzystamy z możliwości

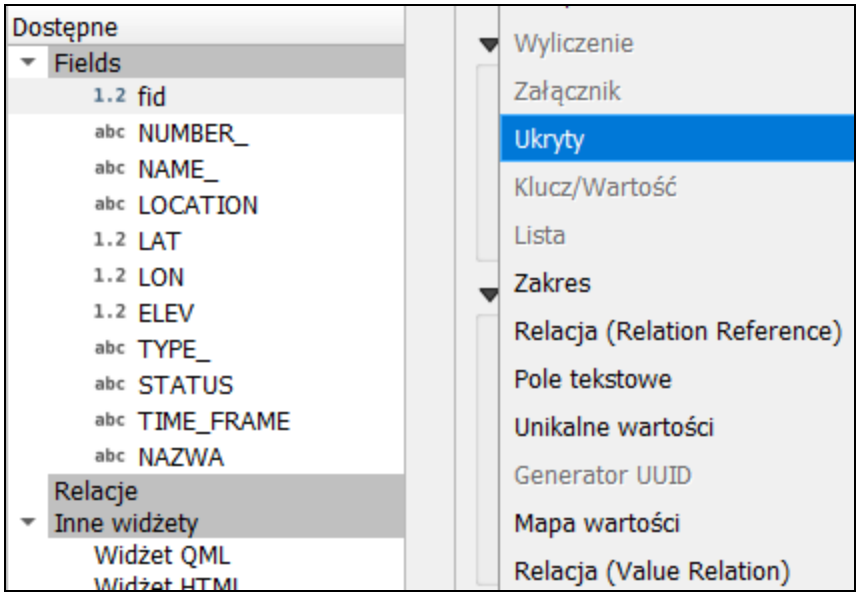
wyłączenia pól z widoku poprzez konfigurację ustawień w oknie *Formularza Atrybutów*. Okno to wywołujemy z poziomu *Właściwości* warstwy *wulkany*:



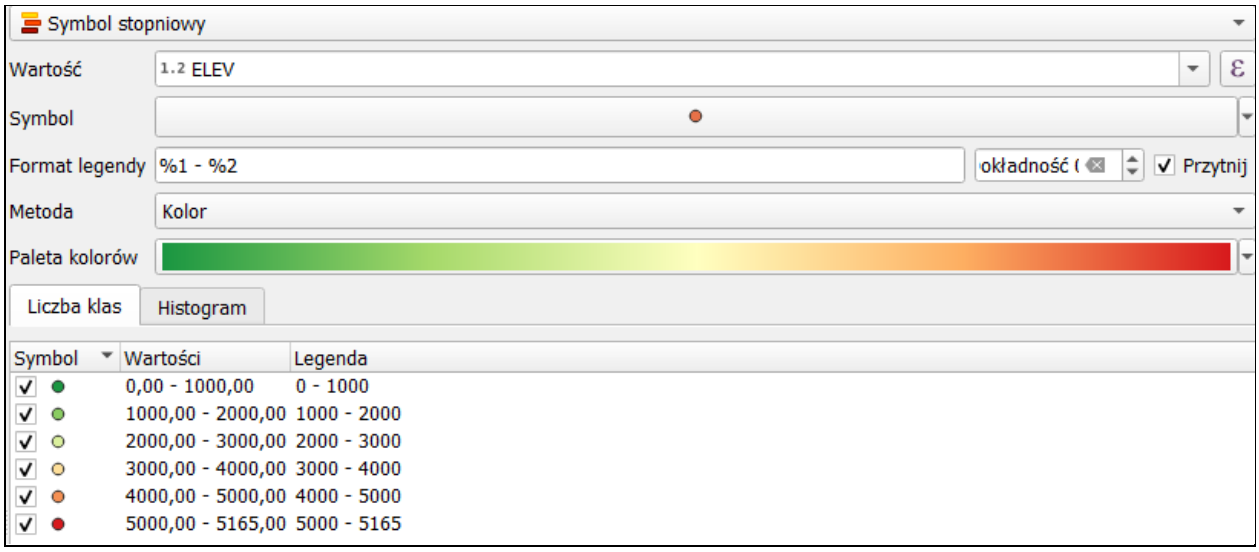
Pierwsze okno po stronie prawej zawiera listę atrybutów warstwy. Wybierzmy atrybut, którego nie chcielibyśmy widzieć na mapie końcowej - przykładowo fid. Po kliknięciu na nim lewym przyciskiem myszy po prawej stronie wyświetli się dodatkowe okno z opcjami. Należy rozwinąć listę *Typ widżetu*



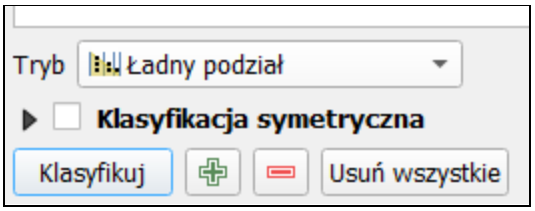
I zamiast *Pole tekstowe* wybrać *Ukryty.* Działanie powtarzamy dla wszystkich atrybutów z wyjątkiem pól NAME_, LOCATION, LAT, LON i ELEV.



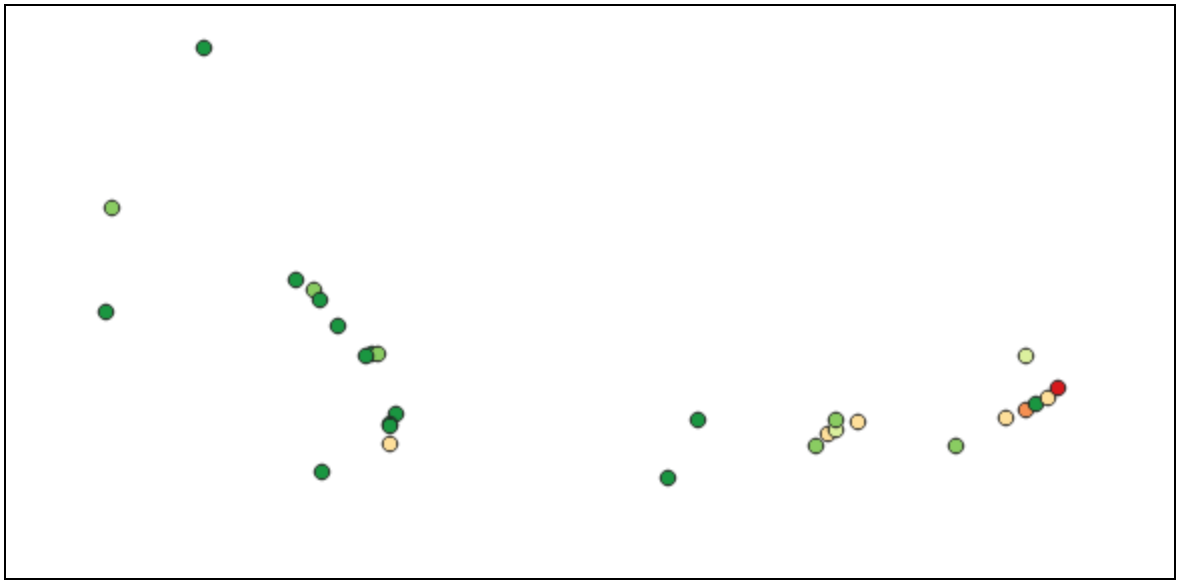
Teraz możemy przystąpić do przygotowania symbolizacji. Docelowo chcielibyśmy uzyskać efekt wizualny, który uwypukli różnice w wysokościach wulkanów. Efekt ten możemy osiągnąć m.in. tworząc symbol stopniowy:



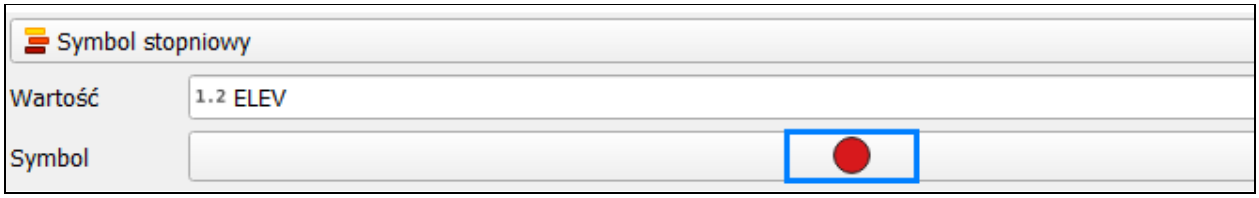
Symbolizację stopniową należy wygenerować na podstawie wartości z pola ELEV. Sugerowana paleta kolorów to RdYlGn (można odwrócić kolejność wyświetlania barw klikając prawym przyciskiem myszy na barwnym pasku i wybierając opcję “odwróć kolory”). Jako tryb klasyfikacji wybieramy “ładny podział”:



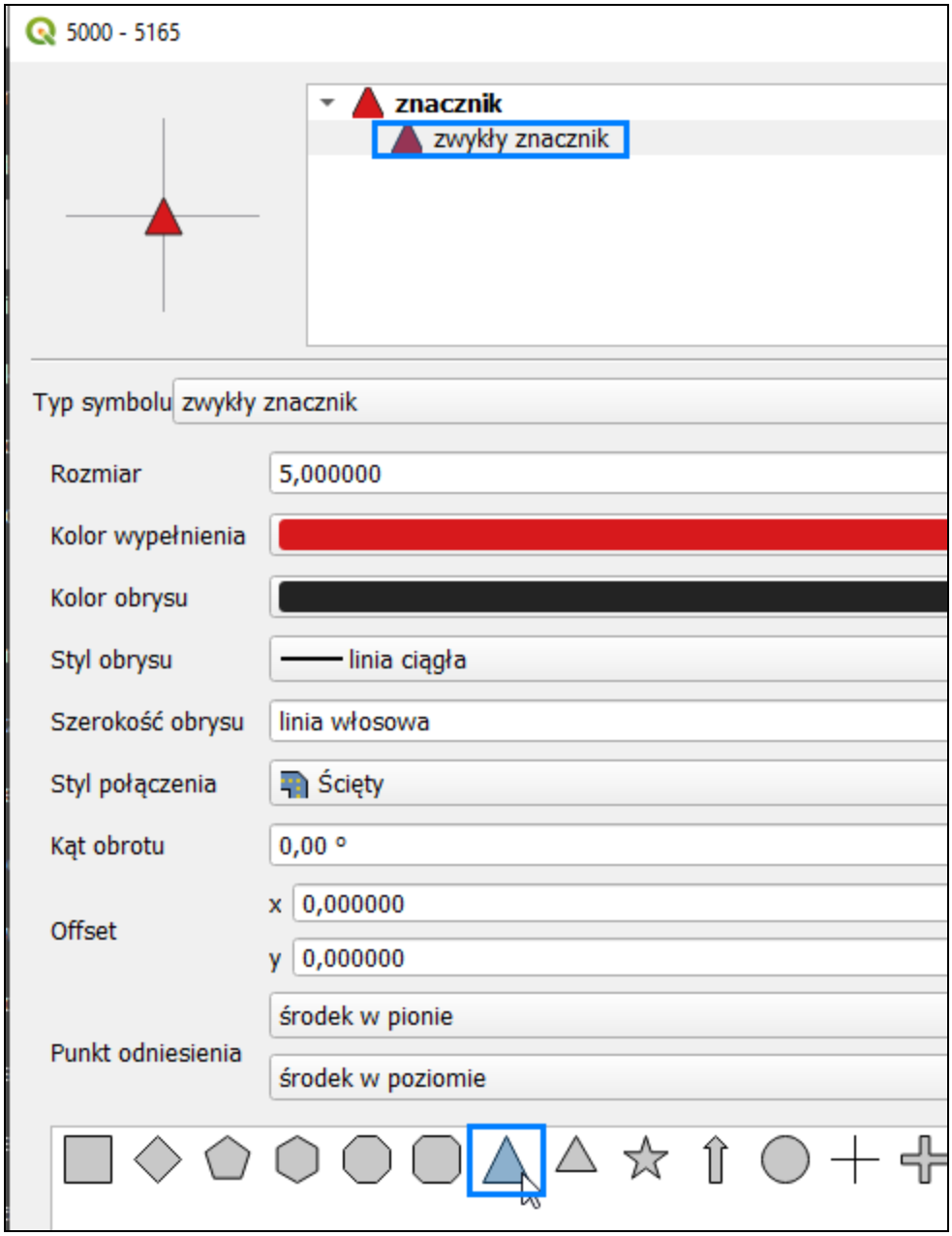
Po zatwierdzeniu ustawień nasza symbolizacja powinna prezentować się następująco:



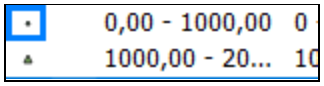
Możemy również zmienić główny znacznik na trójkąt. W tym celu należy w pierwszej kolejności kliknąć na ikonę aktualnego symbolu:

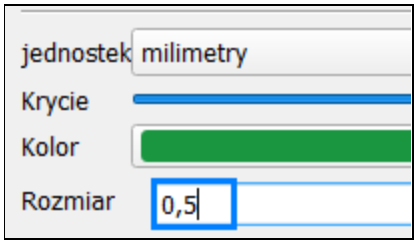


Następnie zaś wybrać nowy znacznik:

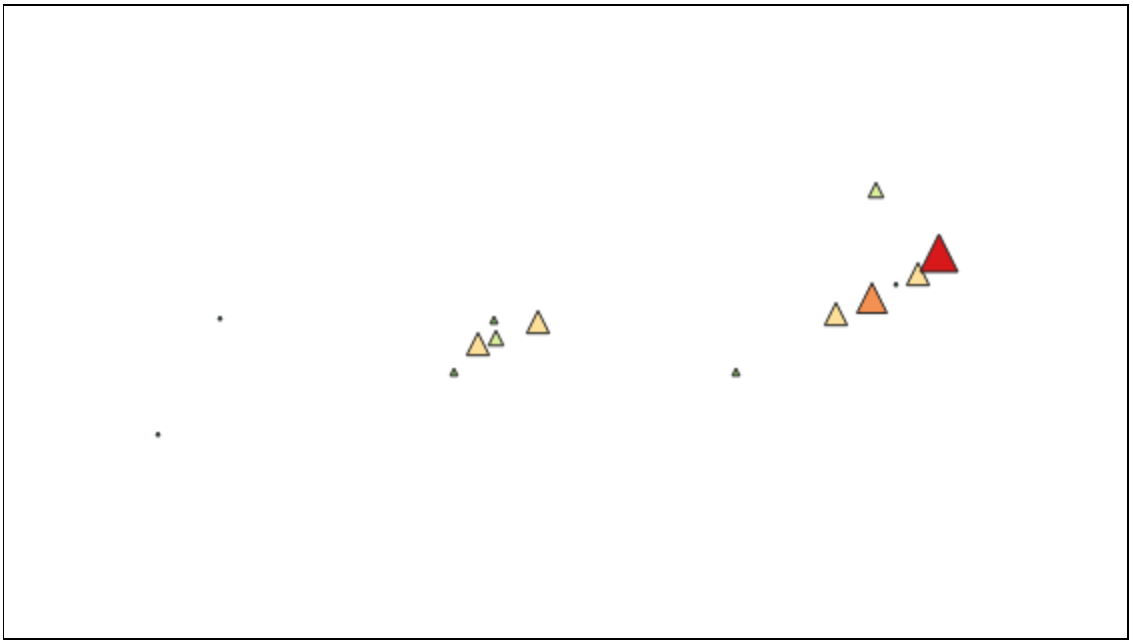


Kolory korespondują z wartościami z *Tabeli Atrybutów*, brakuje natomiast rozróżnienia wielkości symboli. Parametr ten możemy zmodyfikować ręcznie w oknie edycji symbolu. Aby przejść do odpowiedniego okna, należy dwukrotnie kliknąć lewym przyciskiem myszy na symbolu w oknie edycji symbolu, a następnie ustawić parametr:





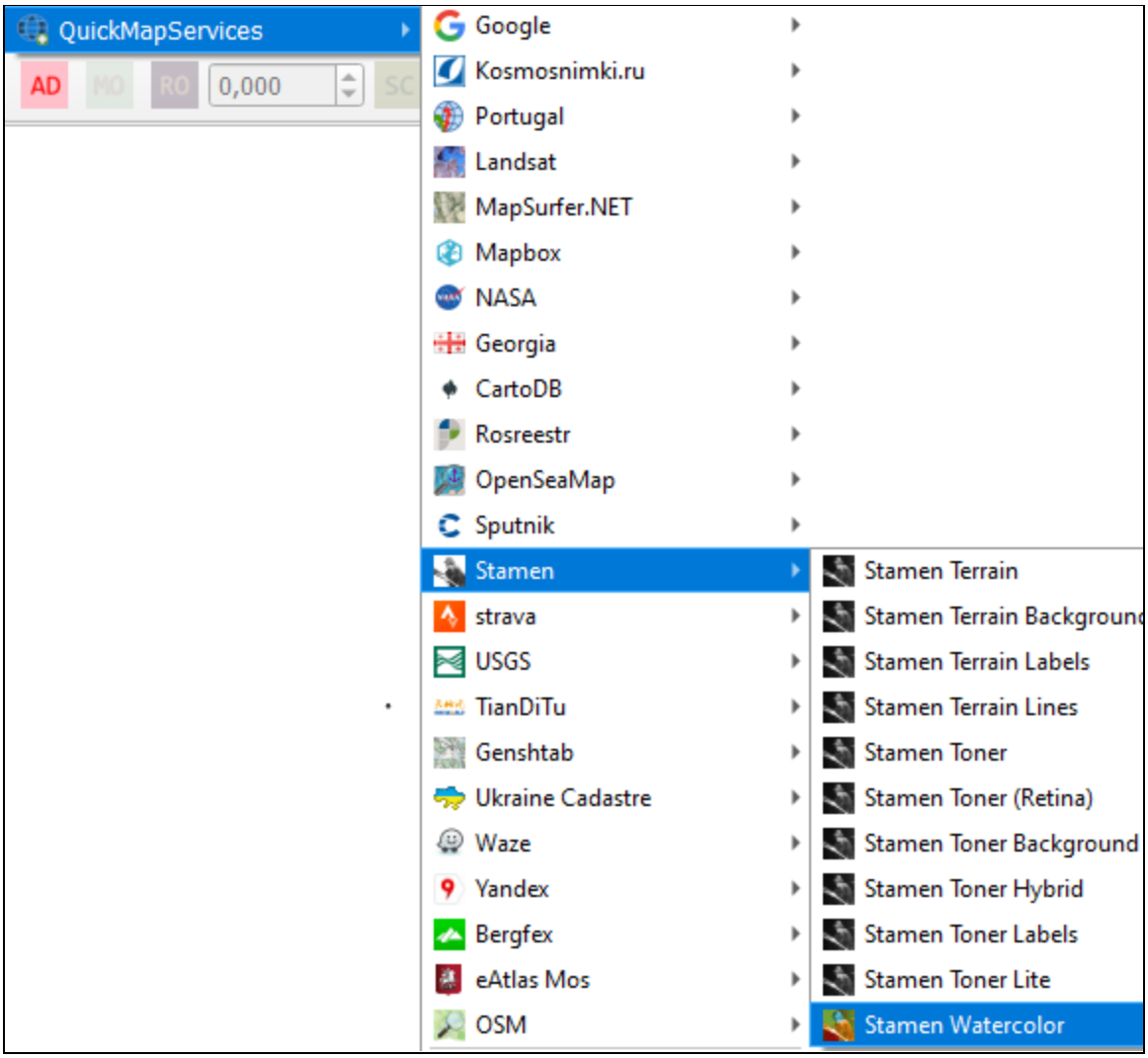
Kolejnej wartości nadajemy rozmiar 1, następnie 2, 3, 4 i 5. Po zatwierdzeniu zmian symbolizacja warstwy powinna prezentować się w sposób zbliżony do przedstawionego poniżej:



Zdecydowanie lepszy efekt wizualny osiągniemy dodając do projektu jeden z gotowych podkładów mapowych. Dostęp do znacznego zbioru predefiniowanych map możemy uzyskać m.in. instalując wtyczkę *quickmapservices*:

Po zainstalowaniu wtyczki w zakładce *W Internecie* na pasku *menu* pojawi się nowa pozycja o nazwie *QuickMapServices*. Rozwijając kolejne zakładki przechodzimy do listy podkładów

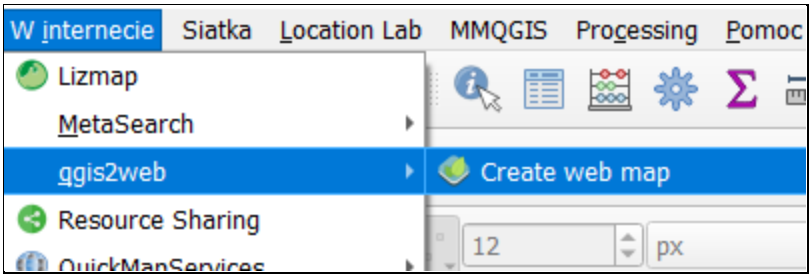
mapowych, które możemy dodać do naszego projektu. Z przygotowaną wcześniej paletą barw bardzo dobrze koresponduje *Stamen Watercolor*:



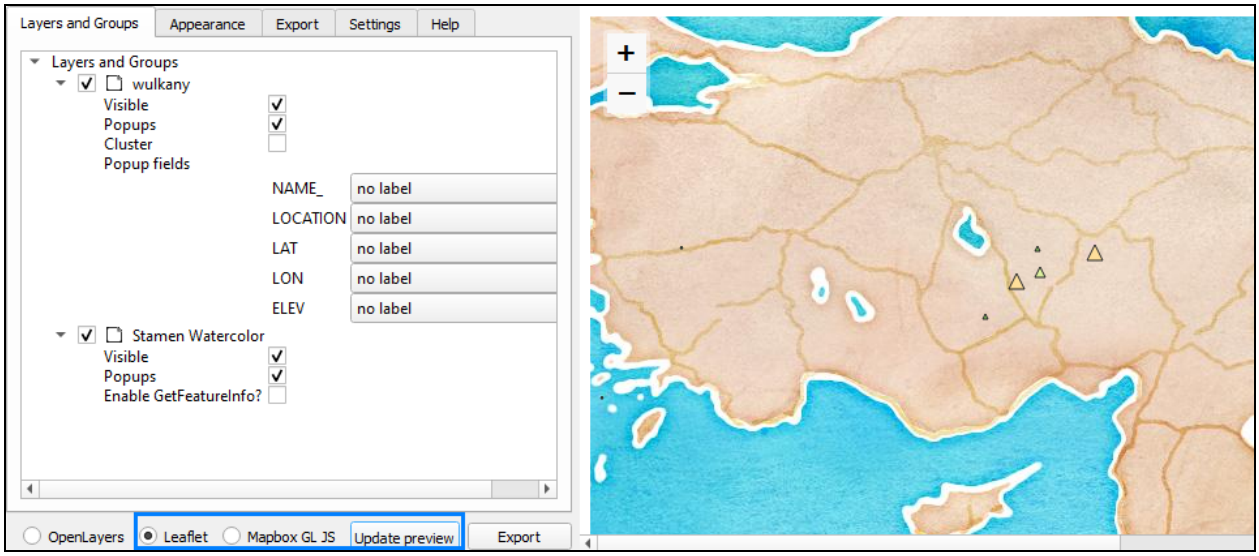
Etap trzeci

Po przygotowaniu symbolizacji możemy przejść do instalacji i konfiguracji wtyczki ***qgis2web***. Wtyczkę instalujemy korzystając z opcji dostępnych w oknie *Zarządzania wtyczkami*:

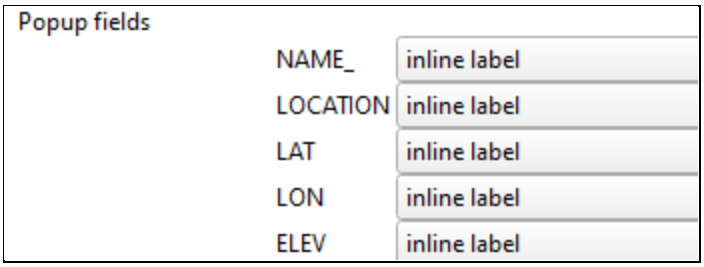
Po instalacji w zakładce *W internecie* pojawi się nowa opcja, którą aktywujemy:



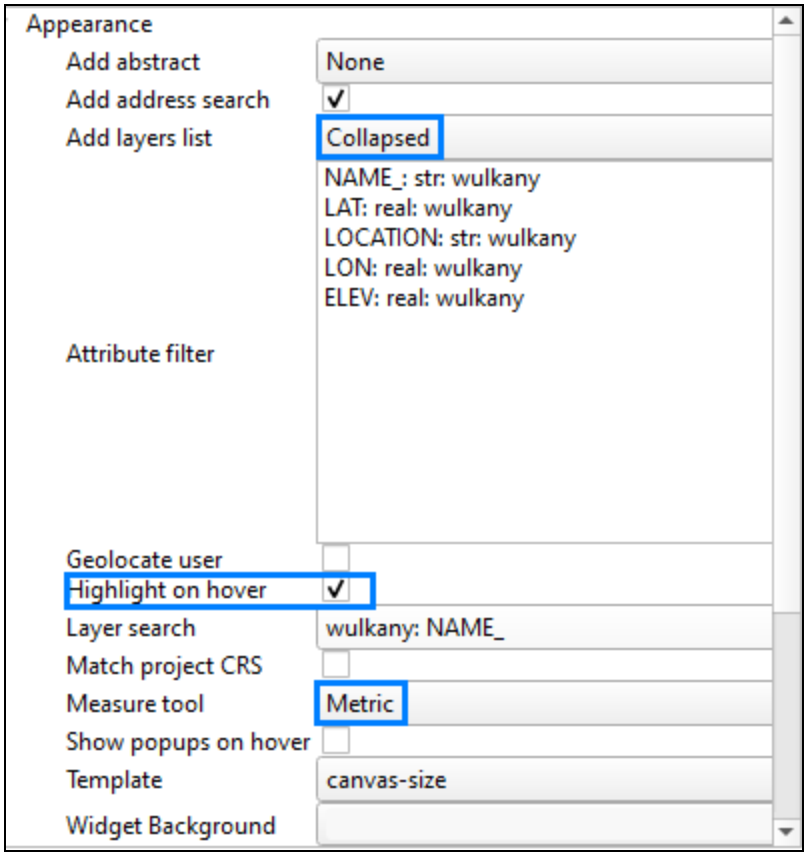
Konfigurację ustawień rozpoczynamy od wyboru biblioteki i odświeżenia widoku projektowanej mapy - efekt ten możemy osiągnąć, klikając na *Update preview* w dolnej części okna wyświetlającego się po lewej stronie:



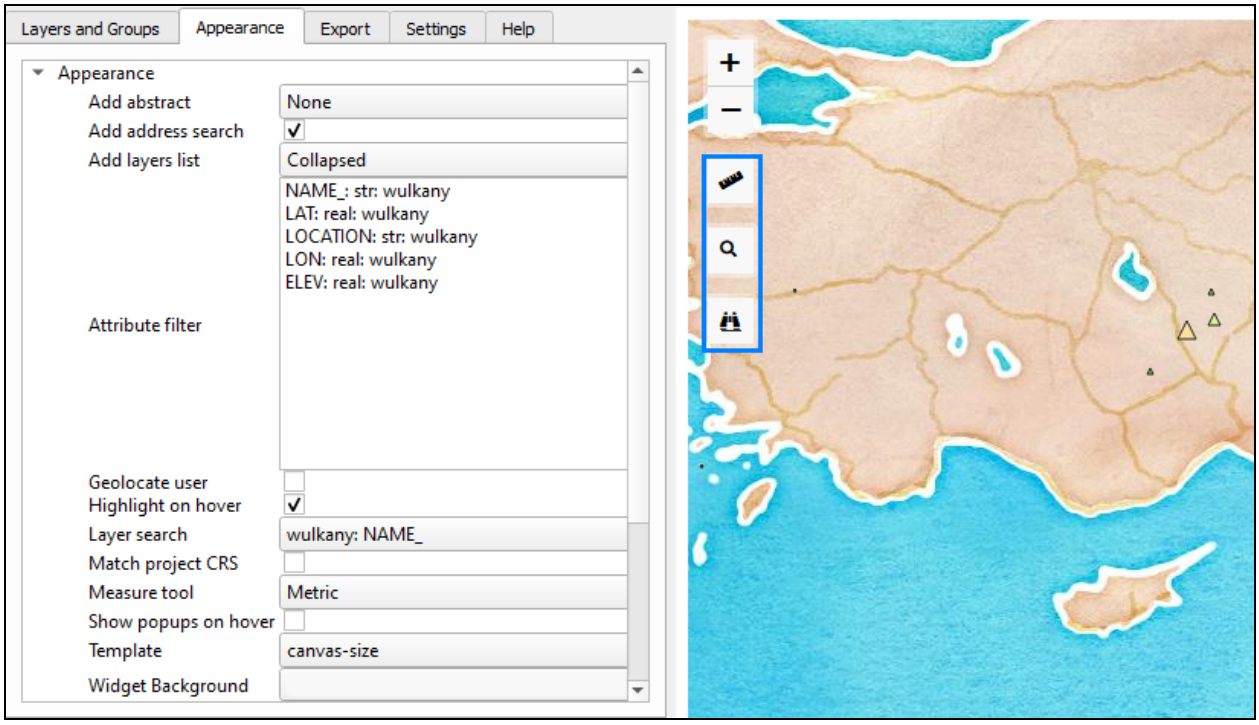
Okno z wizualizacją pozwala nam rozeznać się we wprowadzonych do projektu modyfikacjach. Po każdej zmianie należy ponownie kliknąć na *Update preview*, by odświeżyć widok. Domyślnie po uruchomieniu narzędzia uzyskujemy wgląd w opcje grup i warstw w zakładce *Layers and Groups*. Możemy tu zmienić m.in. ustawienia wyświetlania wartości z *Tabeli Atrybutów*. W tym celu dla każdego pola zamiast *no label* wybieramy *inline label*:



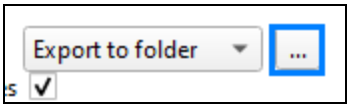
Następnie przechodzimy do zakładki *Appearance*. Z poziomu aktywnego okna możemy dodać do naszej mapy przeglądarkowej narzędzia takie jak wyszukiwarka adresów (*Add address search*), miarkę (*Measure Tool*), czy podświetlenie obiektu po najechaniu na niego kursorem (*Highlight on hover*). Zaznaczając opcję *Collapsed* na liście *Add layers list* sprawimy, że atrybuty wraz z wartościami będą wyświetlać się odrębnym oknie otwierającym się po kliknięciu na wybrany obiekt (wulkan). Niektóre z funkcji, jak np. wyszukiwarka danych atrybutowych, może nie działać poprawnie po wyborze biblioteki *Leaflet*. Pozostałe powinny jednak funkcjonować zgodnie z oczekiwaniami.



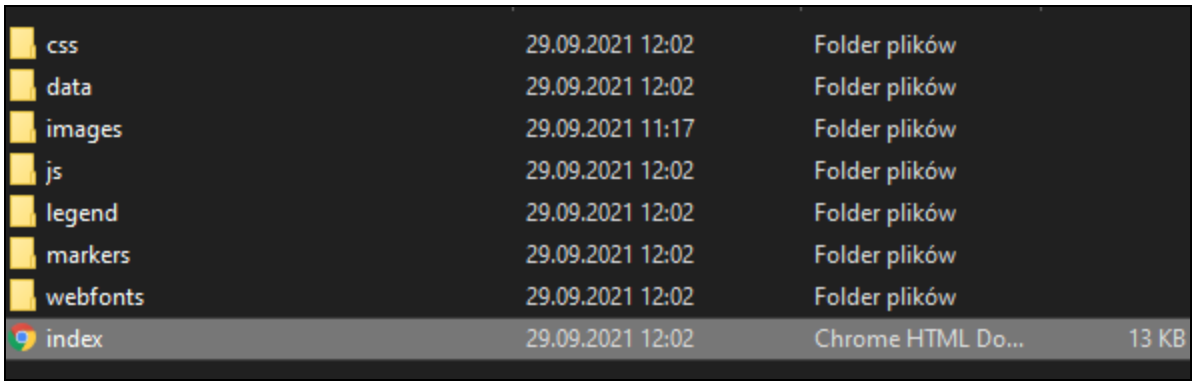
Po ustawieniu parametrów odświeżamy widok i zapoznajemy się z działaniem poszczególnych narzędzi w wyświetlającym się po prawej stronie oknie roboczym.



Tak przygotowaną mapę możemy wyeksportować do wskazanego folderu. W tym celu przechodzimy do zakładki *Export* umiejscowionej, podobnie jak poprzednie, w górnej części głównego okna wtyczki. Katalog docelowy wskazujemy, klikając na ikonę z trzema kropkami. Pozostałe ustawienia możemy pozostawić w trybie domyślnym.



Po wskazaniu katalogu należy kliknąć przycisk *Export*. Wynikowy .html wraz z wszystkimi niezbędnymi plikami powinien pojawić się w podanym folderze:



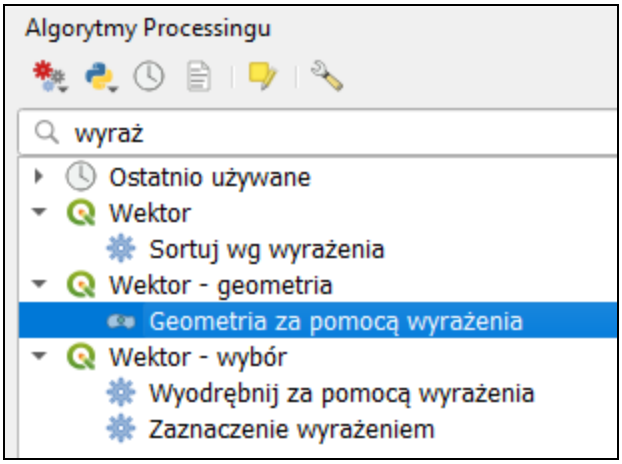
Tak przygotowaną mapę możemy otworzyć w wybranej przeglądarce internetowej. Jeśli dysponujemy dostępem do zdalnego serwera, możemy przenieść na niego nasze pliki i udostępniać mapę innym użytkownikom.

Zaawansowane analizy przestrzenne z wykorzystaniem narzędzia DB Manager

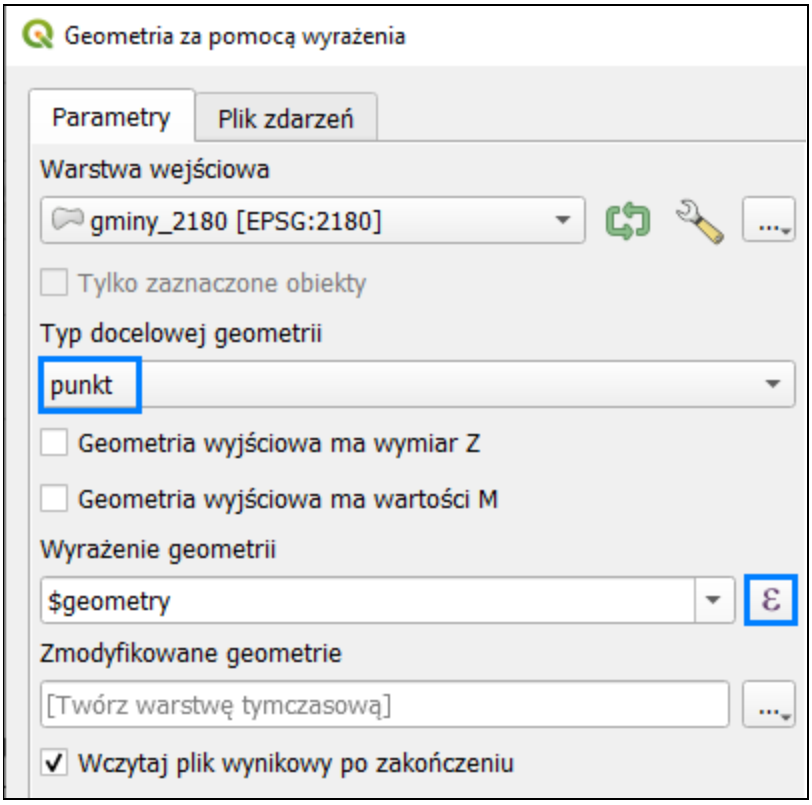
Zadanie polega na wygenerowaniu pięciu stref “dostaw” wokół punktu centralnego (centroid utworzony dla granic Polski) o szerokości 12, 25 i 52 i 90 km i numeracji od I do IV. Następnie wykorzystując iloczyn przestrzenny należy przypisać informację o strefie do obiektów poligonowych reprezentujących gminy. Efektem końcowym powinna być wizualizacja strefowości gmin, wykorzystująca symbolizację unikalną. Do dyspozycji mamy warstwę poligonową gminy_2180.

Zaczynamy od utworzenia centroidu dla granic Polski. Naturalnie w pierwszej kolejności należy wygenerować zagregowany obiekt ze wszystkich poligonów warstwy gminy_2180.

Możemy skorzystać z narzędzia *Geometria za pomocą wyrażenia*:



W oknie dialogowym algorytmu należy wskazać warstwę wejściową, typ geometrii wynikowej oraz formułę generującą wynik. Tę ostatnią można wpisać w oknie kreatora, które wywołujemy klikając na zaznaczoną na obrazku ikonę:



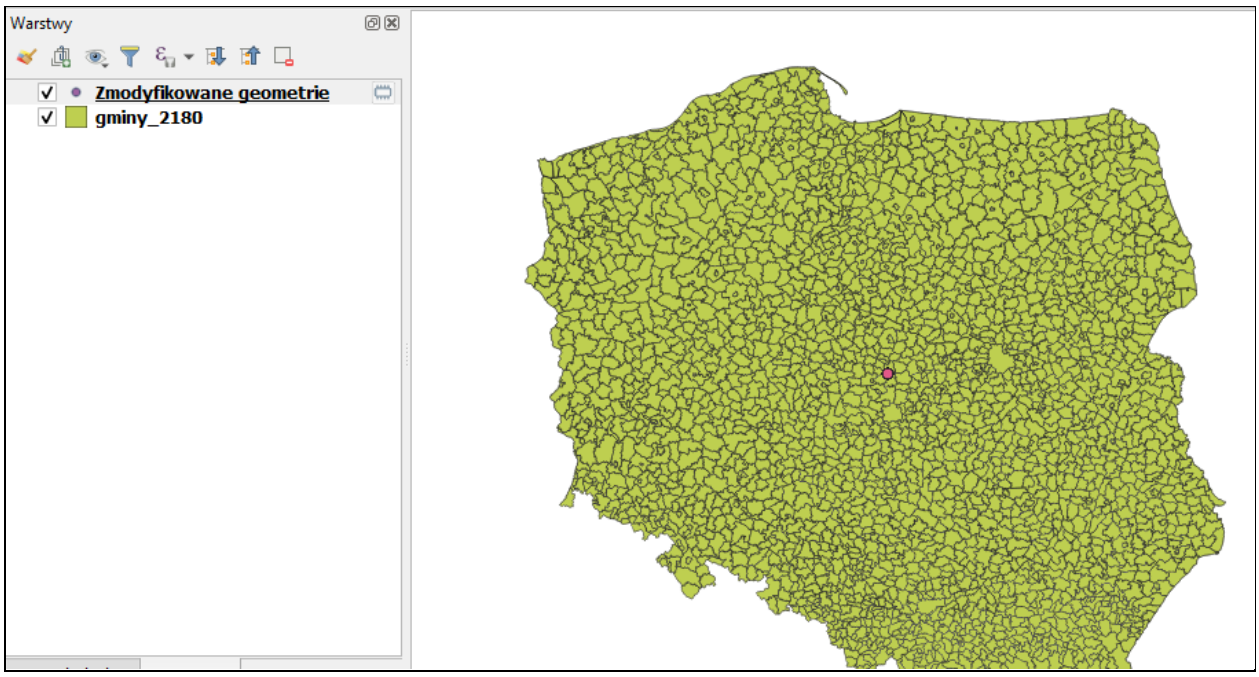
Aby uzyskać centroid granic kraju, musimy najpierw dokonać połączenia wszystkich gmin w jeden obiekt. Możemy osiągnąć ten efekt korzystając z formuły:

aggregate('gminy_2180', collect, $geometry)

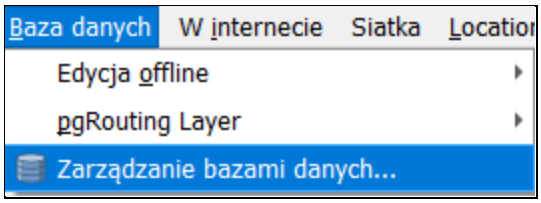
"Zbierze" ona wszystkie geometrie składowe, dzięki czemu w kolejnym kroku będziemy mogli wygenerować centroid:

centroid(aggregate('gminy_2180', collect, $geometry)))

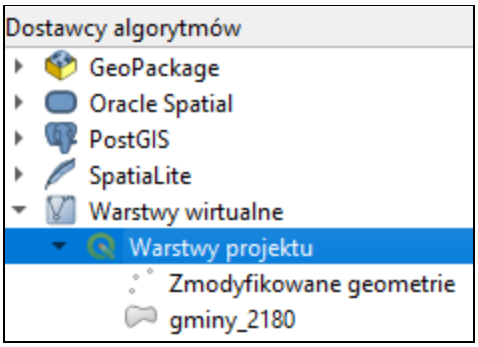
Po zatwierdzeniu formuły wracamy do głównego okna algorytmu i klikamy na *Uruchom*. UWAGA! Proces przetwarzania może zająć nawet kilka minut! W wyniku otrzymamy warstwę o nazwie *Zmodyfikowane geometrie*, która zawiera oczekiwany punkt środkowy:



Do wygenerowania stref buforowych wykorzystamy narzędzie *Zarządzanie Bazami Danych* z zakładki *Bazy danych*:



Po otwarciu okna rozwijamy zakładkę *Warstwy wirtualne*, a następnie *Warstwy projektu*:



Tworzymy nową kwerendę, klikając na ikonę  umieszczoną w prawym, górnym rogu okna. Dysponujemy już gotowym centroidem w postaci warstwy *Zmodyfikowane geometrie*. Nazwa jest jednak dość długa, przez co jej ustawiczne powtarzanie w instrukcji SQL może być niewygodne. Spróbujmy więc na wstępie zdefiniować ją jako zmienną:

```
WITH
cn AS
(SELECT DISTINCT geometry  AS geom FROM
"Zmodyfikowane geometrie")
```

Następnie przejdźmy do utworzenia strefy I o zasięgu 12 km:

```
SELECT ST_buffer(cn.geom,12000,90) AS geom
FROM cn
```

A następnie dołączmy do niego pierwszy pierścień w postaci różnicy przestrzennej buforów 25 i 12 km:

```
UNION
SELECT ST_Difference(ST_buffer(cn.geom,25000,90),
ST_buffer(cn.geom,12000,90)) AS geom
FROM cn
```

Oraz kolejne obiekty. Pełne wyrażenie generujące bufor wraz z pierścieniami powinno wyglądać następująco:

```
SELECT
ST_Buffer(cn.geom, 12000, 90) AS geom
FROM cn
UNION
```

```
SELECT
ST_Difference(ST_Buffer(cn.geom, 25000, 90),
ST_Buffer(cn.geom, 12000, 90)) AS geom
FROM cn
UNION
SELECT
ST_Difference(ST_Buffer(cn.geom, 52000, 90),
ST_Buffer(cn.geom, 25000, 90)) AS geom
FROM cn
UNION
SELECT
ST_Difference(ST_Buffer(cn.geom, 90000, 90),
ST_Buffer(cn.geom, 52000, 90)) AS geom
FROM cn
```

Całe to wyrażenie również możemy wziąć w nawias i potraktować jako zmienną *buf*. Istnieje jednak drobny problem; wykonujemy działania na warstwach wirtualnych, przez co nie mamy dostępu do wielu przydatnych funkcji PostgreSQL, jak choćby dodawania kolumny z numerem porządkowym. Musimy zatem wymyślić sposób, aby przyporządkować naszym poligonom oznaczenia stref od I do IV. Jednym z pomysłów jest wykorzystanie pomiaru powierzchni - im większa wartość, tym wyższy nr strefy. Dla ułatwienia wynik można przedstawić w kilometrach i zaokrąglić do liczby całkowitej.

```
SELECT round(ST_Area(str.geom)/1000000) AS pow,
str.geom AS geom
FROM str
ORDER BY pow
```

	pow	geom
1	452	Polygon ((5287...
2	1511	Polygon ((5417...
3	6531	Polygon ((5687...
4	16951	Polygon ((6067...

Mając wgląd w liczby możemy stworzyć wyrażenie warunkowe, które wygeneruje kolumnę o nazwie strefy i wypełni ją wartościami od I do IV:

```
SELECT round(ST_Area(buf.geom)/1000000) AS pow,
CASE
WHEN round(ST_Area(buf.geom)/1000000) <= 1000 THEN 'I'
```

WHEN round(ST_Area(buf.geom)/1000000)<= 2000 THEN 'II'
WHEN round(ST_Area(buf.geom)/1000000) <= 6600 THEN 'III'
ELSE 'IV'
END AS strefa,
buf.geom AS geom
FROM buf
ORDER BY pow

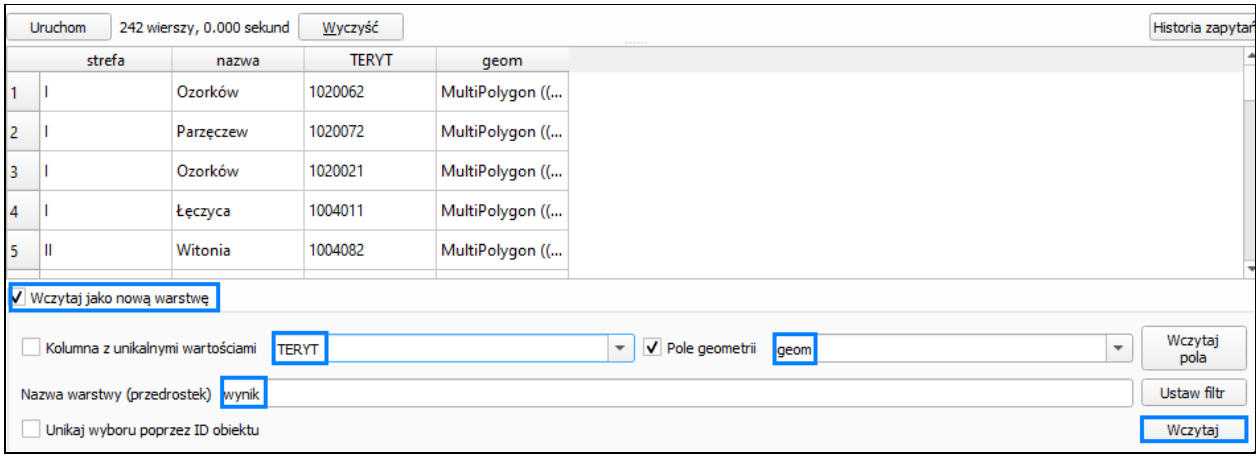
Uruchom | 4 wierszy, 0.000 sekund | Wyczyść

	pow	strefa	geom
1	452	I	Polygon ((5287...
2	1511	II	Polygon ((5417...
3	6531	III	Polygon ((5687...
4	16951	IV	Polygon ((6067...

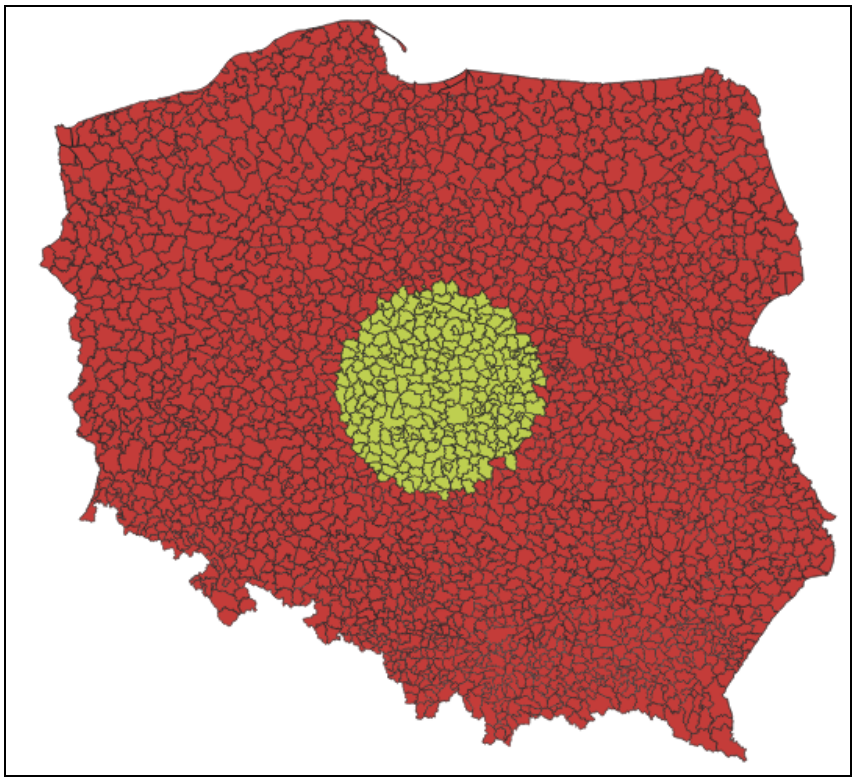
Powoli przechodzimy do ostatniej części ćwiczenia, tj. wykorzystania stref do przygotowania zestawienia gmin mieszczących się w ich zasięgu. Przyjmijmy, że głównym kryterium przynależności gminy do strefy jest wartość nakładania się większa niż 50%. Dla ułatwienia z poprzedniego wyrażenia utworzymy zmienną *strefy*, którą wykorzystamy w kolejnej instrukcji:

SELECT a.strefa, b."JPT_NAZWA_" AS nazwa,
b."JPT_KOD_JE" as TERYT,
b.geometry AS geom
FROM
strefy a, gminy_2180 b
WHERE
ST_Intersects(a.geom,b.geometry) AND
ST_Area(ST_Intersection(a.geom, b.geometry))/ST_Area(b.geometry)*100>50

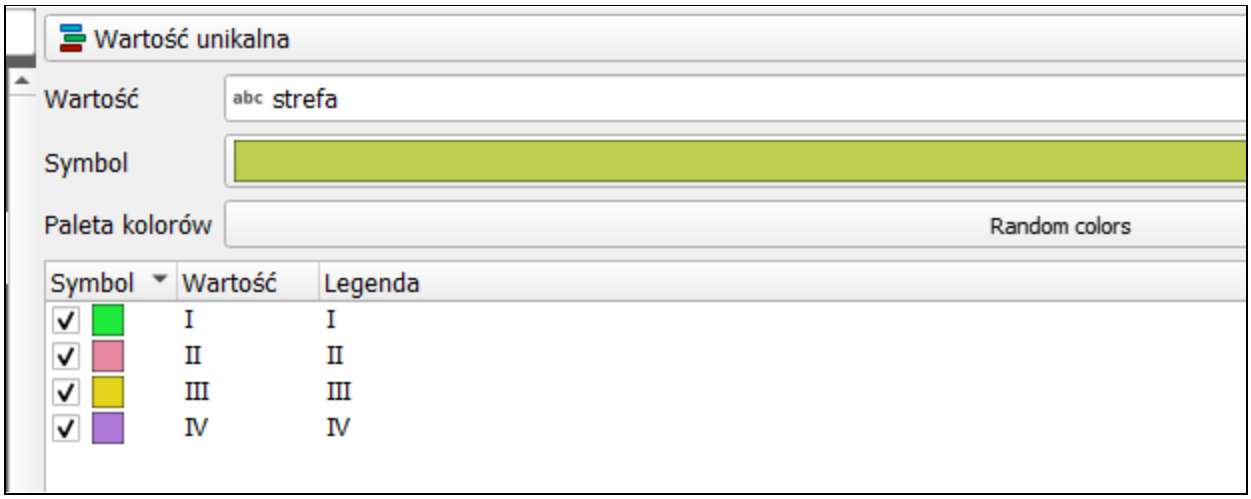
W efekcie powinniśmy uzyskać 242 geometrie wynikowe. Dodajmy wynik kwerendy jako nową warstwę przestrzenną:



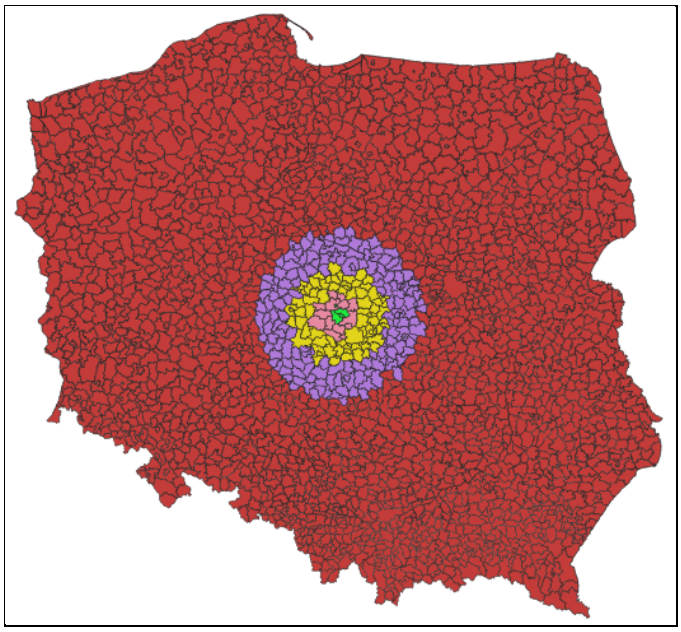
Wynik w surowej, przestrzennej wersji prezentuje się następująco:



Pozostaje nam więc ostatni szlif, tj. stworzenie symbolizacji unikalnej na podstawie wydzielonych stref:



A oto rezultat końcowy:

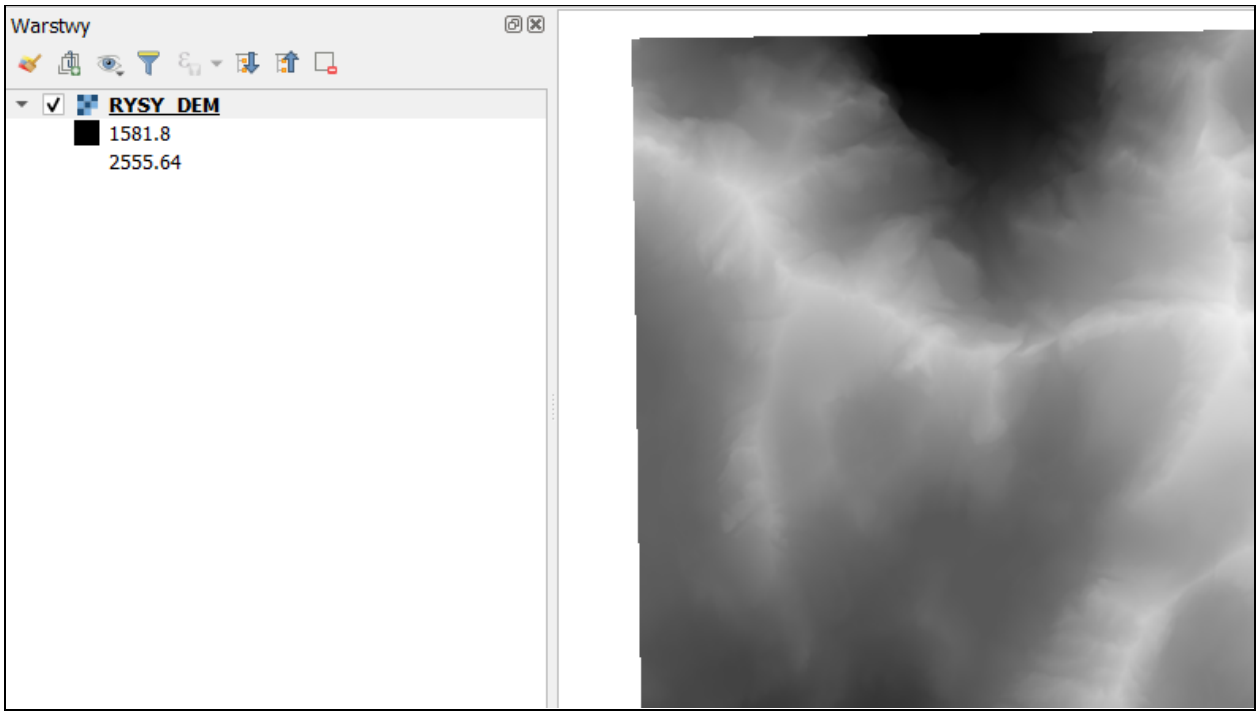


Przykładowa analiza wykorzystująca dane z Numerycznego Modelu Terenu (np. wyszukiwanie terenów o określonych parametrach spadku i ekspozycji)

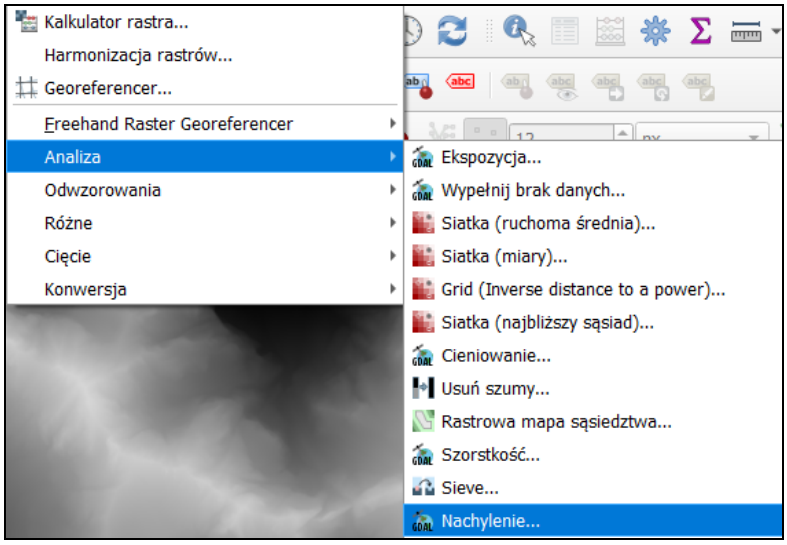
Analiza terenu

Celem ćwiczenia jest wygenerowanie mapy spadków i ekspozycji, a następnie reklasyfikacja wartości jednego z rastrów wynikowych. Efektem końcowym działań będzie warstwa przedstawiająca rozmieszczenie stoków górskich o wystawie południowej. W kolejnym kroku spróbujemy przygotować trójwymiarową wizualizację terenu i sprawdzić, czy ekspozycja została obliczona poprawnie.

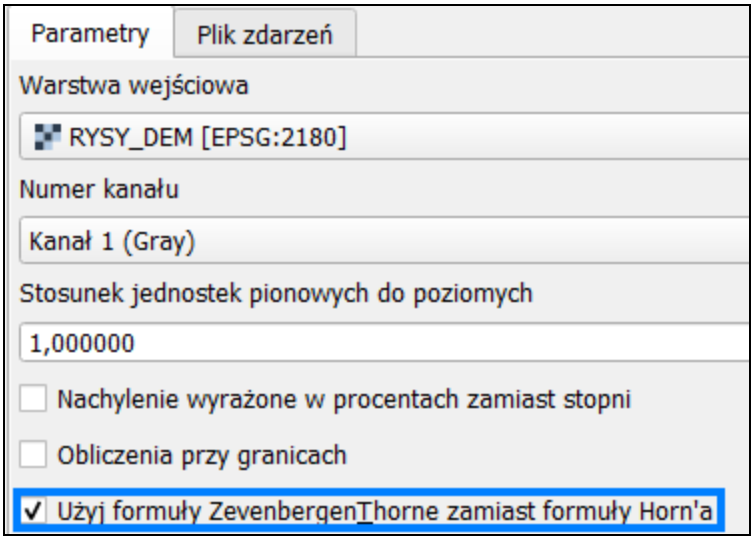
Tworzymy nowy projekt, nadajemy mu układ współrzędnych o kodzie EPSG:2180 i dodajemy warstwę rastrową RYSY_DEM.tif (można ją przeciągnąć z katalogu do okna programu):



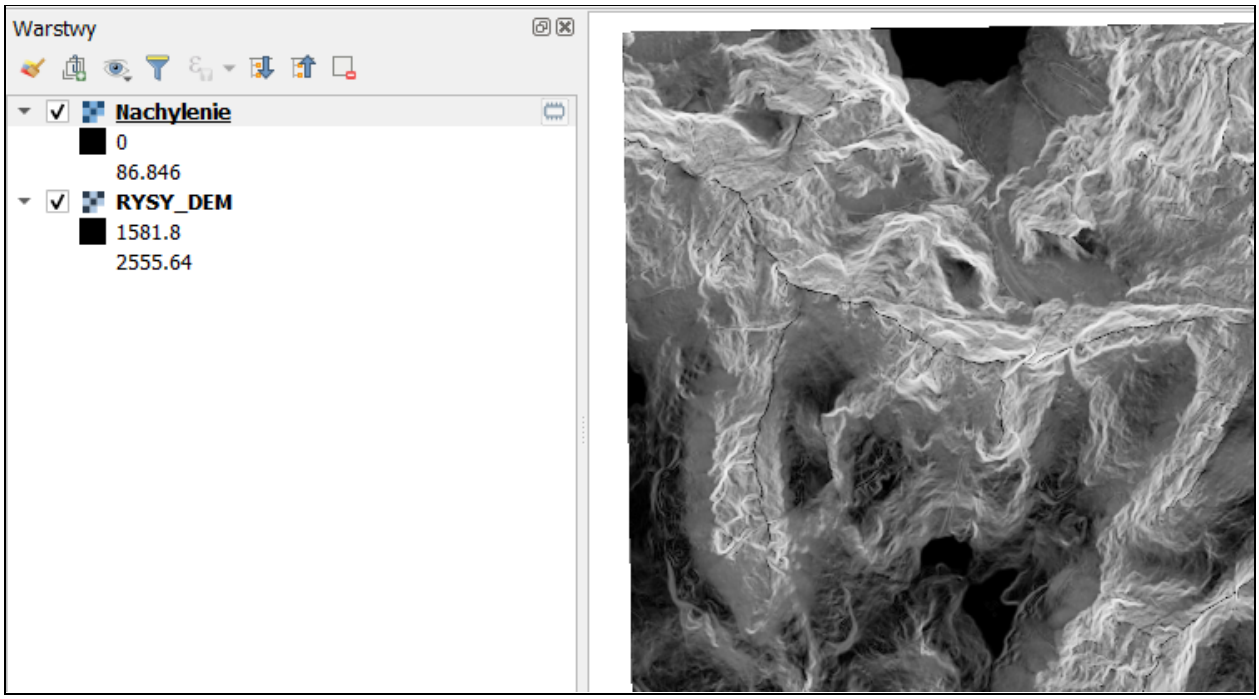
Aby wygenerować mapę spadków, przechodzimy do zakładki *Raster -> Analiza* i wybieramy *Nachylenie*:



W oknie algorytmu wskazujemy warstwę źródłową (RYSY_DEM). Większość parametrów możemy pozostawić bez zmian. Jedyna modyfikacja dotyczy użycia formuły *ZevenbergenThorne* w miejsce domyślnej *Horne'a* (zabieg ten pozwoli uzyskać bardziej wygładzoną, a przez to estetyczniejszą reprezentację terenu).

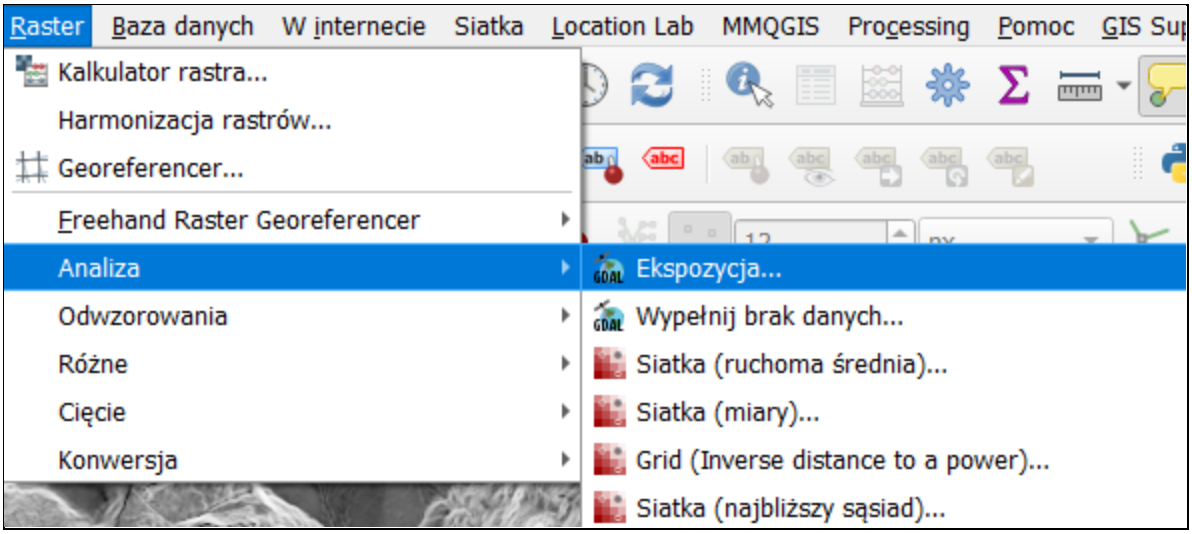


Po wprowadzeniu zmian klikamy na *Uruchom*. Surowa wizualizacja rastra wynikowego przedstawia się następująco:

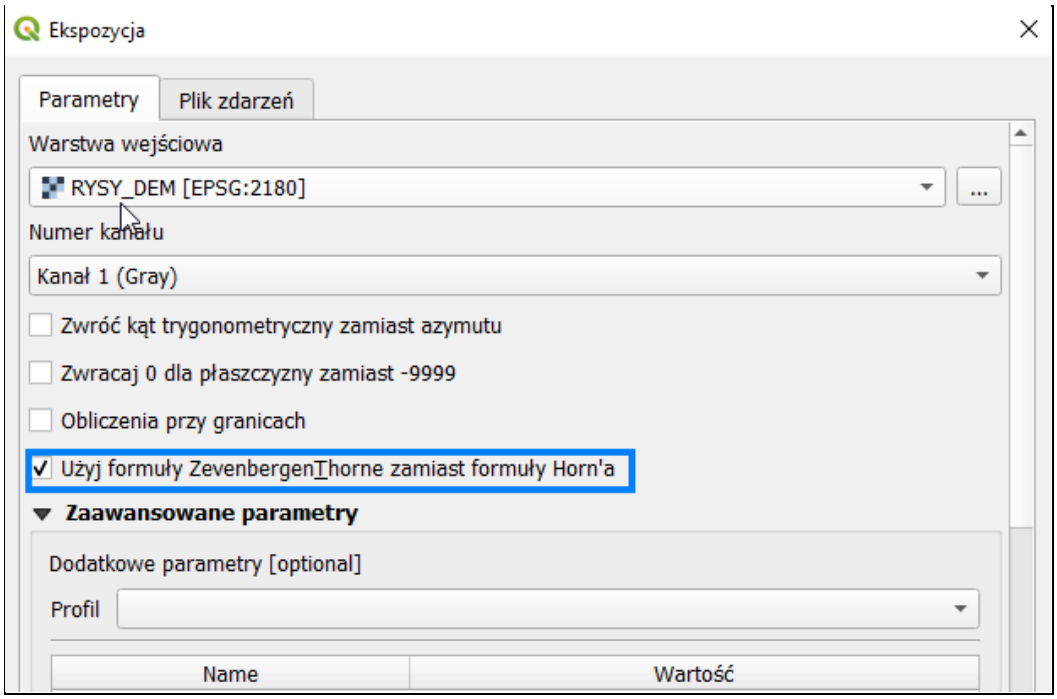


Przy domyślnym doborze barw kolor czarny oznacza obszary o zerowym bądź bardzo małym nachyleniu; wraz ze wzrostem tej wartości barwa przechodzi z czarnej w szarą, na koniec zaś w białą. Kolor biały reprezentuje więc obszary o największym nachyleniu.

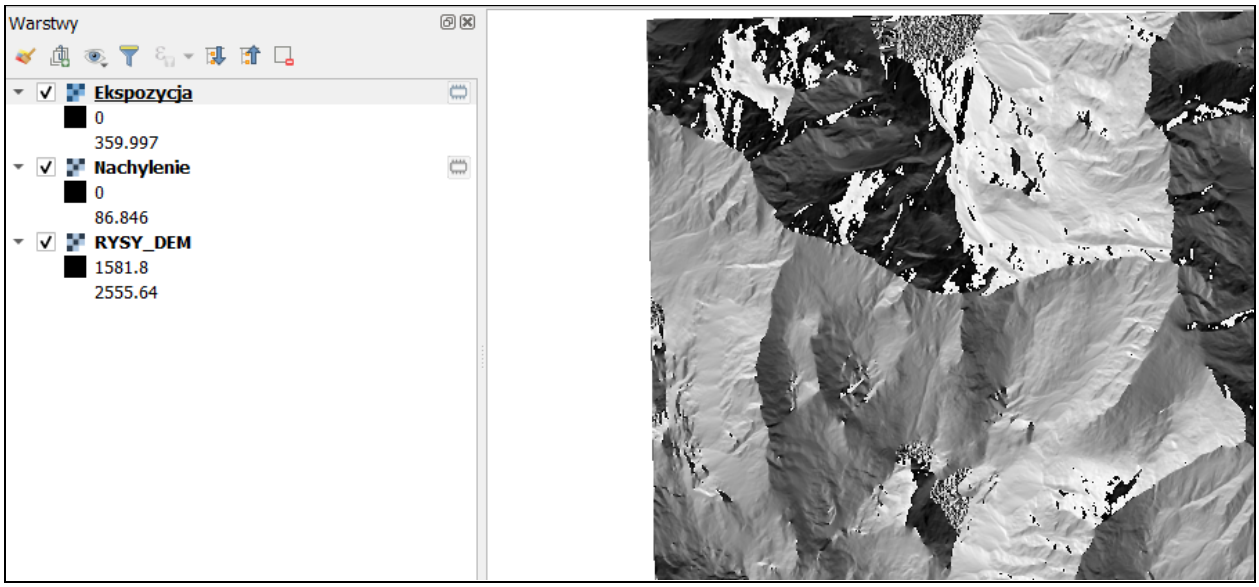
Spróbujmy teraz wygenerować mapę ekspozycji. Odpowiedni algorytm ponownie znajdziemy w zakładce *Raster -> Analiza* (szukamy opcji *Ekspozycja*):



Parametry dobieramy analogicznie do mapy nachyleń:

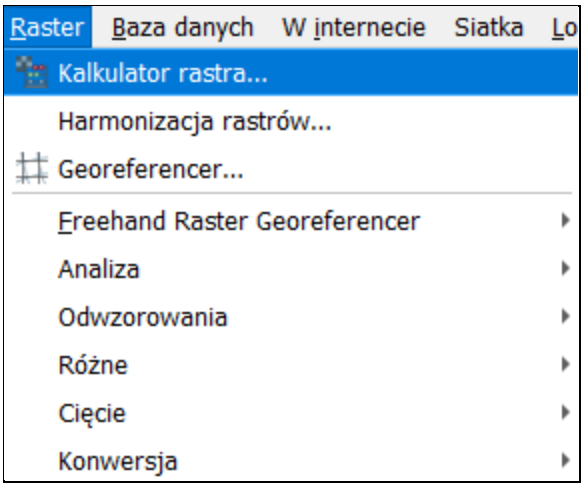


Po kliknięciu na *Uruchom* możemy zbadać obraz wynikowy. Domyślna wizualizacja nie jest zbyt czytelna:



Sposobem na zwiększenie czytelności mapy ekspozycji jest reklasyfikacja rastra. Piksele, które w przypadku warstw omawianego typu przyjmują wartości od 0 do 360 (stopnie), można pogrupować w kategorie reprezentujące poszczególne warianty wystawy terenu. Do przeprowadzenia poprawnej reklasyfikacji konieczna jest znajomość wartości liczbowej wyznaczającej kierunek północny. W programie QGIS jest to 0. Wartości rosną wraz z ruchem

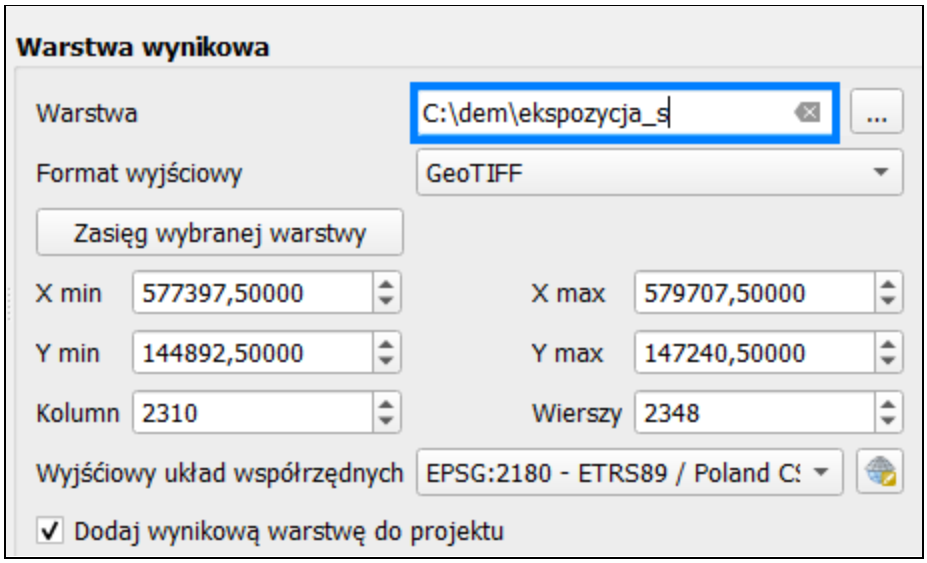
wskazówek zegara: kierunek wschodni przypada na 90 stopni, południowy 180, zachodni 270. Nasze zadanie polega na wizualizacji stoków południowych. Dla lepszego zobrazowania wybranej ekspozycji posłużymy się wycinkiem koła o przedziale 135 - 225 stopni. Do wykonania reklasyfikacji wykorzystamy *Kalkulator rastra*, znajdujący się w zakładce *Raster*:



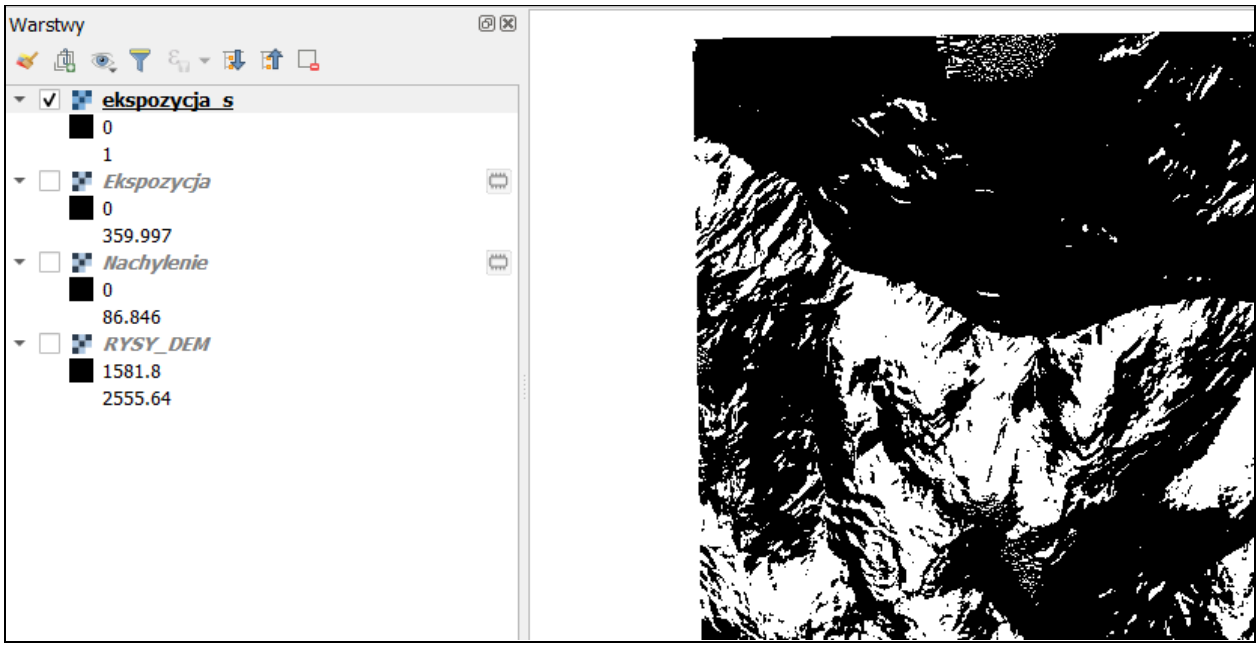
W widoku kalkulatora znajduje się lista dostępnych warstw oraz okno, w którym należy wpisać wyrażenie reklasyfikujące. Interesują nas jedynie wartości z przedziału 135-225, co można zapisać następująco:

```
"Ekspozycja@1">=135 AND "Ekspozycja@1"<=225
```

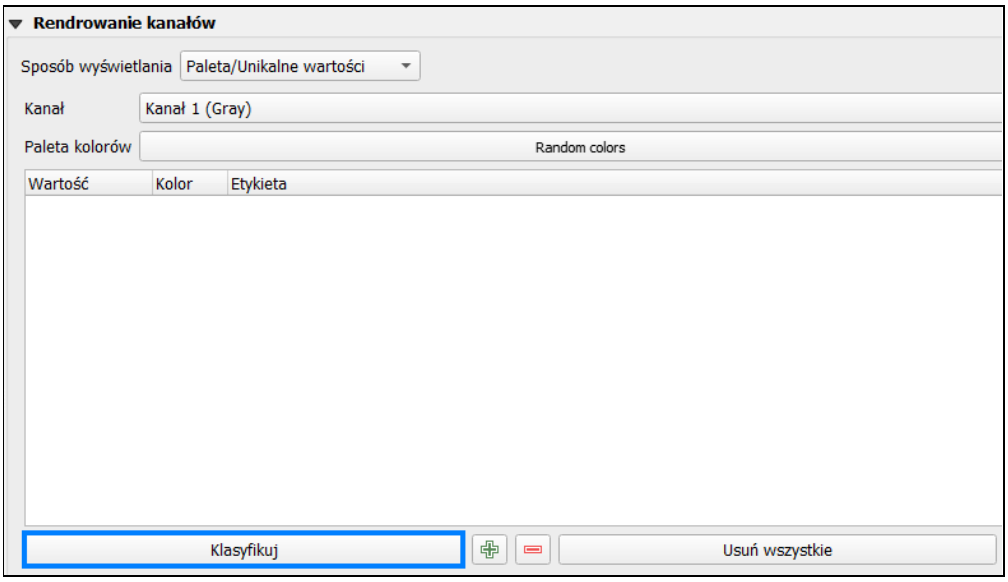
Dodatkowo należy zdefiniować ścieżkę zapisu pliku wynikowego:



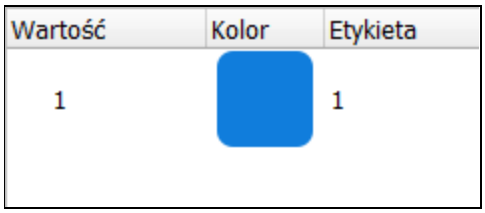
Po wprowadzeniu wszystkich ustawień klikamy na *OK* i przechodzimy do okna mapy. Widok powinien przedstawiać się następująco:



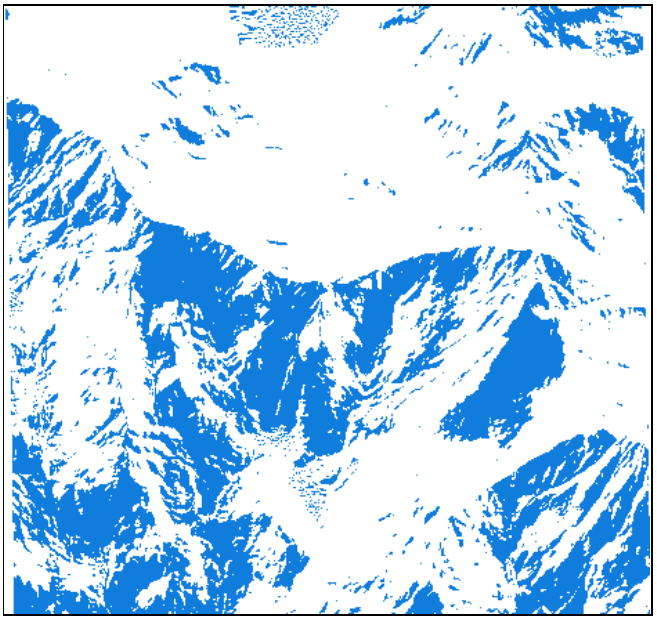
Możemy poprawić efekt końcowy modyfikując obraz tak, by wyświetlały się jedynie piksele o wartości 1. W tym celu przechodzimy do właściwości warstwy *ekspozycja_s* i rozwijamy zakładkę *Styl*. Z listy dostępnych trybów wizualizacji wybieramy *Paleta/unikalne wartości*. Klikamy na *Klasyfikuj*:



Zaznaczamy wartość 0 i usuwamy ją, klikając na ikonę  . Możemy również zmodyfikować kolor przyporządkowany wartości 1:

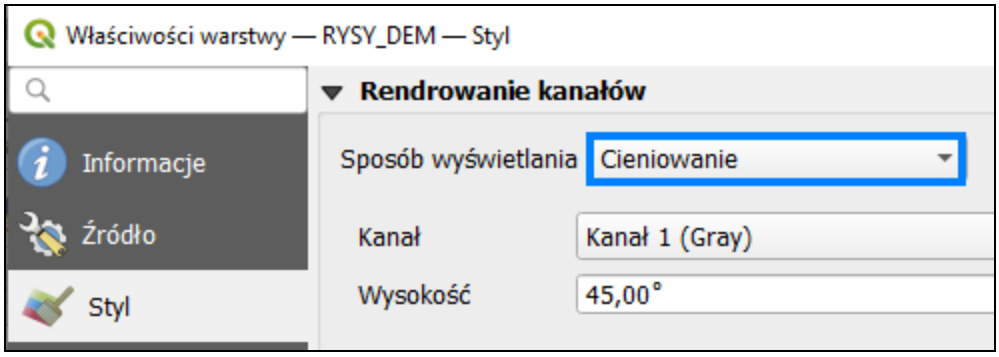


Po wprowadzonych zmianach obraz wynikowy powinien prezentować się w sposób zbliżony do tego przedstawionego na ilustracji:

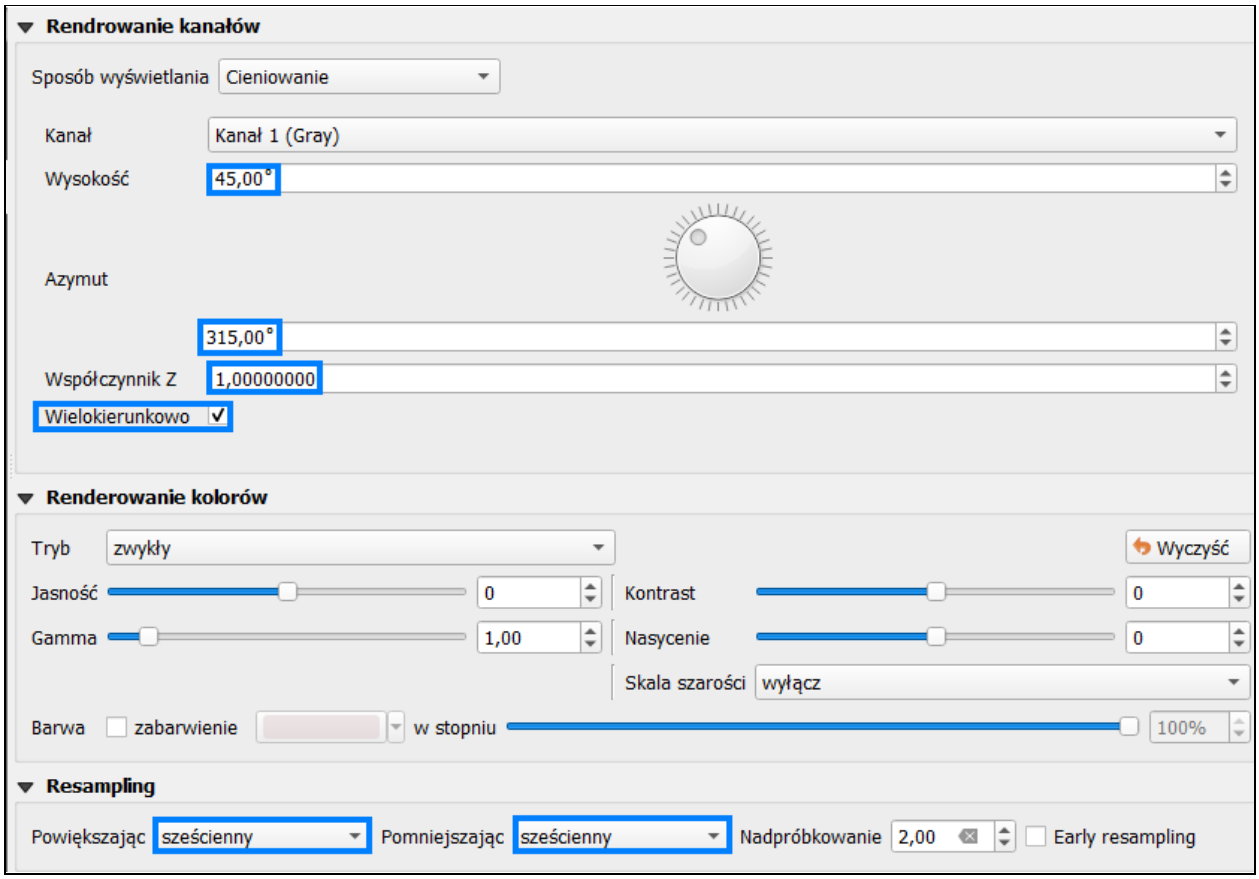


Wizualizacja danych NMT w 3D

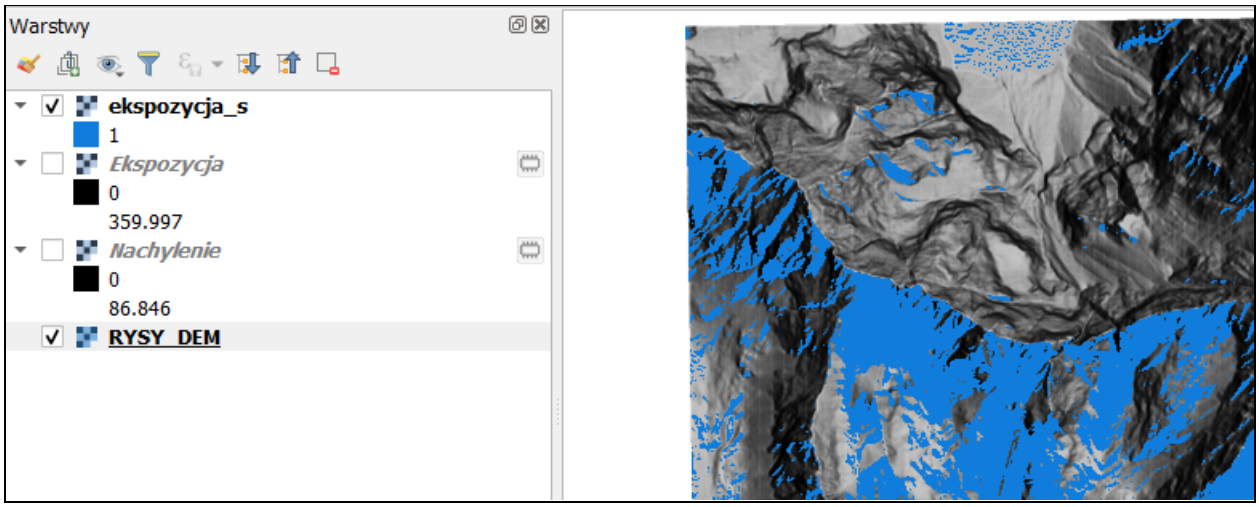
W naszym projekcie znajduje się warstwa rastrowa RYSY_DEM, którą możemy wykorzystać do przygotowania trójwymiarowej wizualizacji terenu. Dla lepszego zobrazowania ukształtowania terenu zmienimy sposób jej wizualizacji. Przechodzimy do właściwości warstwy, wybieramy zakładkę *Styl* i z listy dostępnych opcji wybieramy *Cieniowanie*:



Wybrany sposób renderowania daje nam możliwość ustawienia parametrów zbliżonych do tych, które definiujemy w oknie algorytmu *Cieniowanie (hillshade)*. Możemy więc określić wysokość źródła światła (im większa wartość, tym mocniej naświetlony, a przez to jaśniejszy obraz wynikowy), kierunek oświetlenia (domyślnie pn-zach, co odpowiada wartości 315 stopni) oraz wartość przewyższenia rzeźby. Dodatkowo wprowadzono opcję *wielokierunkowo* - zaznaczając ją sprawimy, że teren będzie oświetlany z czterech kierunków określonych wartościami 225, 270, 315 i 360 stopni. Modyfikacja ustawień resamplingu w dolnej części okna spowoduje natomiast, że obraz wynikowy uzyska bardziej wygładzony wygląd. Parametry ustawiamy według wzoru przedstawionego na poniższej ilustracji:

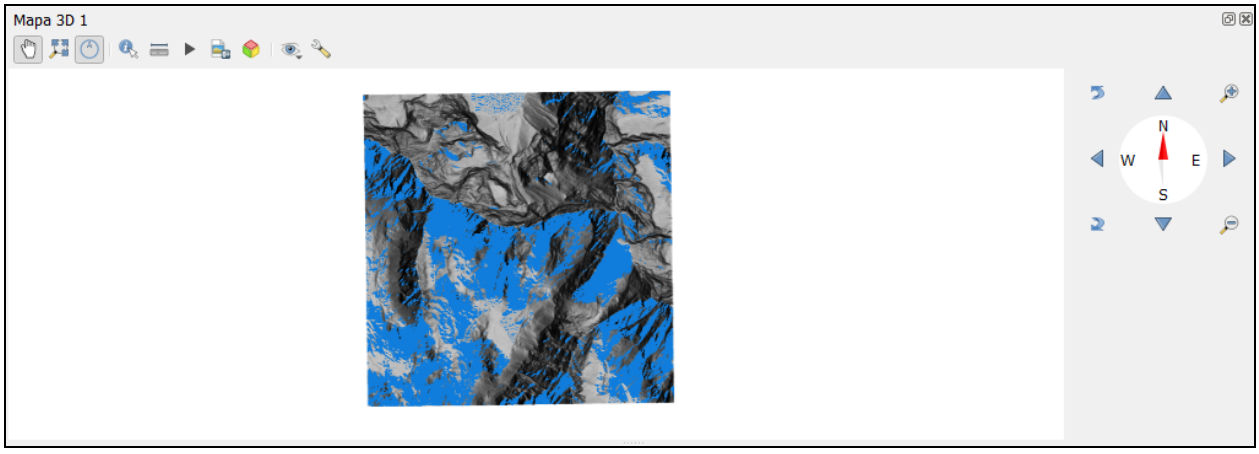


Aby wprowadzić zmiany klikamy na *Zastosuj*, a następnie na *OK*. Na tym etapie wizualizacja powinna wyglądać tak:



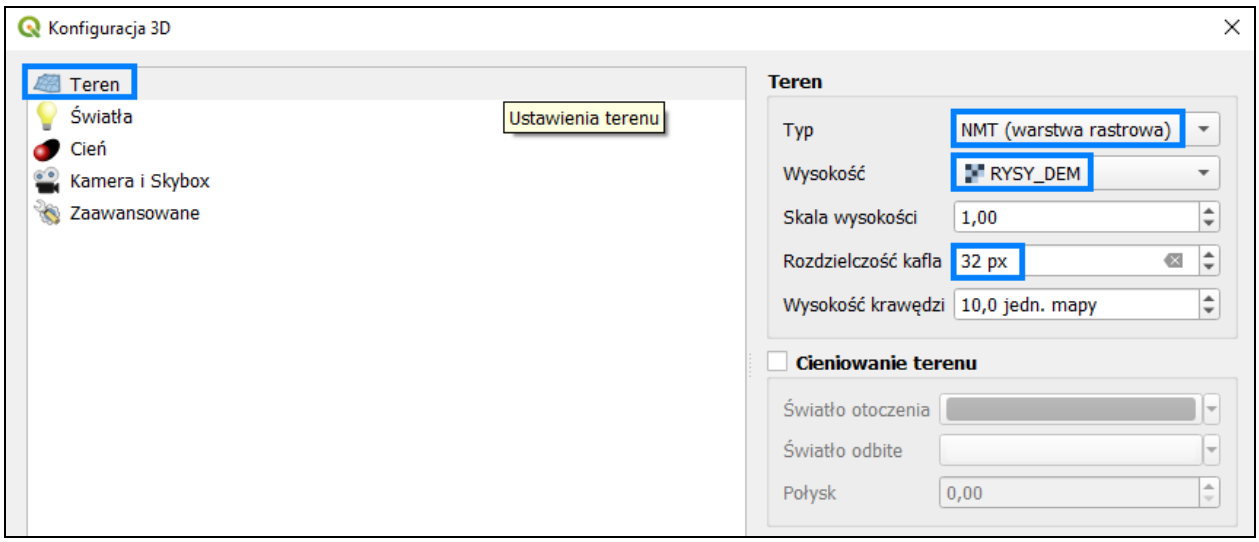
Możemy przejść do przygotowania wizualizacji 3D. W tym celu rozwijamy zakładkę *Widok* na pasku *menu* i wybieramy *Nowy widok 3d mapy*:

Otworzy się nowe, dokowalne okno, które możemy umieścić m.in. w górnej części widoku mapy:

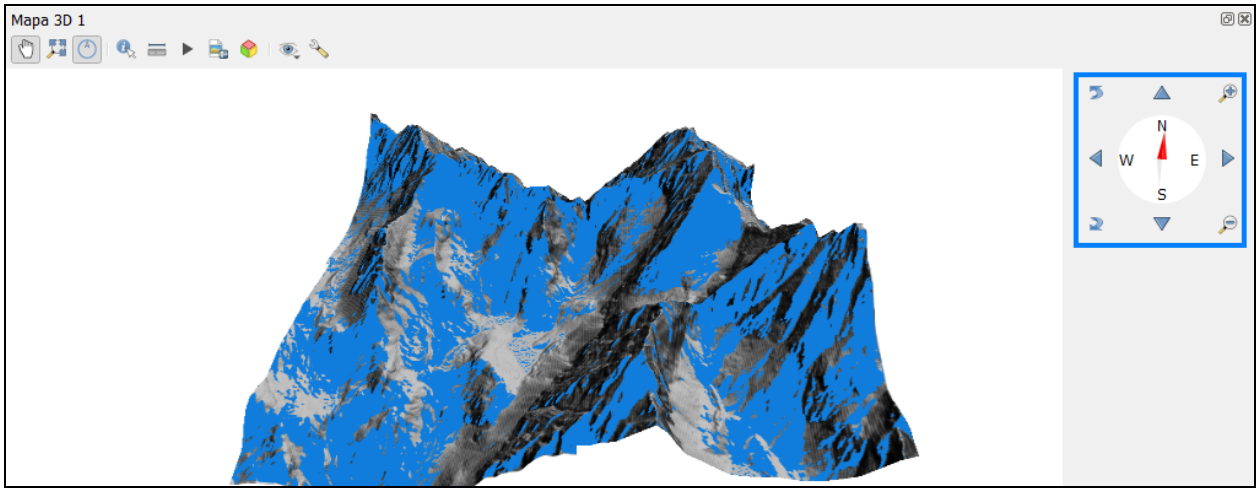


Do wygenerowania wizualizacji 3D niezbędna jest modyfikacja bazowych ustawień. Odpowiedni panel wywołujemy klikając na ikonę klucza na pasku narzędzi:

W kolejnym oknie wybieramy zakładkę *Teren*, następnie w ustawieniach *Terenu* po prawej stronie określamy typ (NMT warstwa rastrowa) oraz wskazujemy warstwę z danymi wysokościowymi (RYSY_DEM). Dodatkowo możemy nieco zwiększyć rozdzielczość kafla (poprawi to jakość obrazu wynikowego).



Klikamy na *Zastosuj*, *OK* i wracamy do widoku mapy 3D. Możemy zmienić położenie widoku, trzymając jednocześnie wciśnięte klawisz SHIFT i lewy przycisk myszy oraz przesuwając kursor w wybranym kierunku. Dodatkowo po prawej stronie znajduje się kompas ułatwiający orientację w trakcie nawigacji:



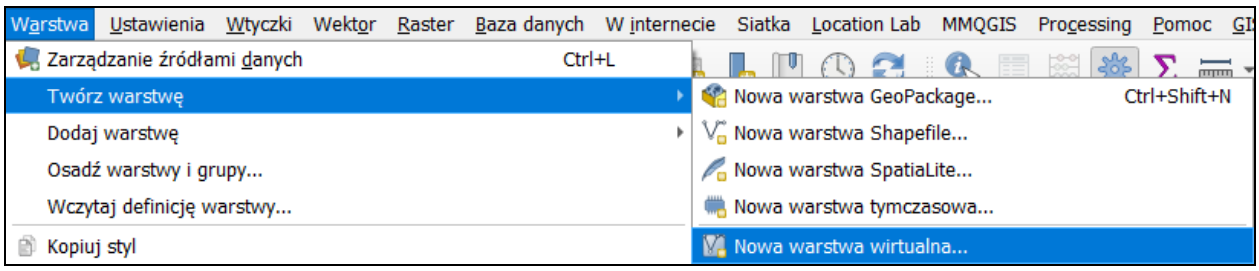
Ćwiczenie dodatkowe - generowanie szlaków na podstawie NMT

Pierwsza część ćwiczenia polega na wyznaczeniu najkrótszej drogi z trzech punktów startowych 1,2 i 3 do punktu docelowego. Znane są współrzędne punktu końcowego:

578584.1,145954.9

Przyjmujemy ponadto, że wszystkie punkty mają znajdować się w takiej samej odległości od punktu centralnego, tj. 950 m. Przy tworzeniu geometrii posłużymy się warstwą wirtualną oraz algorytmem *Geometria za pomocą wyrażenia*.

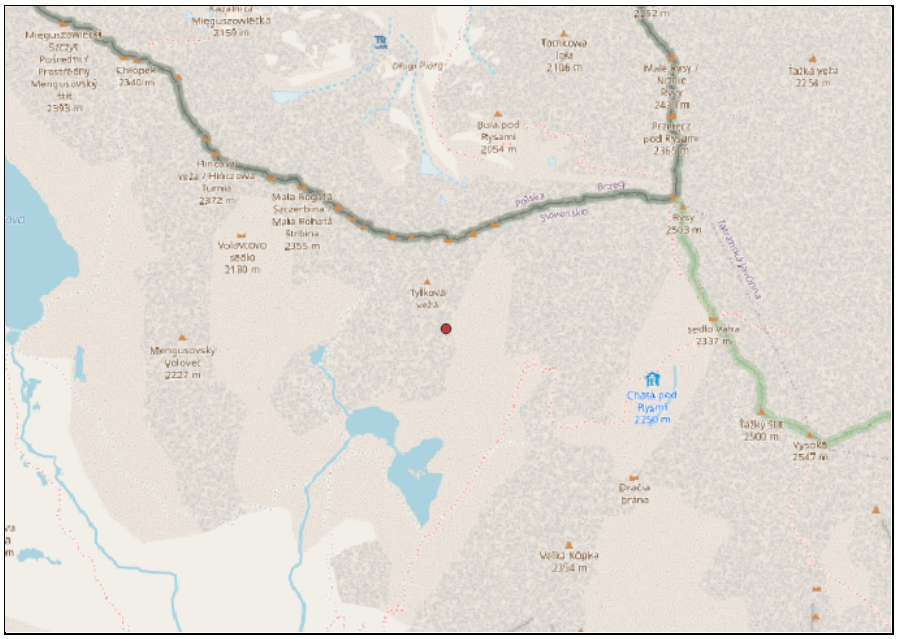
Zacznijmy od stworzenia punktu o podanych współrzędnych. Rozwijamy zakładkę *Warstwa -> Twórz warstwę* i wybieramy



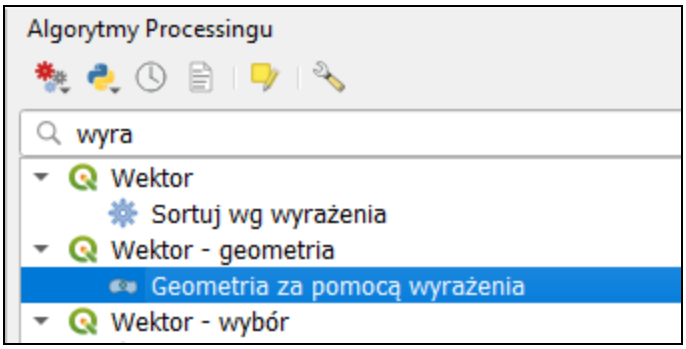
W oknie kreatora warstwy wirtualnej zmieniamy nazwę warstwy oraz wprowadzamy instrukcję generującą punkt o podanych wyżej koordynatach. Definiujemy ponadto układ współrzędnych przy pomocy funkcji ***SetSrid()***.

SELECT SetSrid(make_point(578584.1,145954.9),2180) AS geom

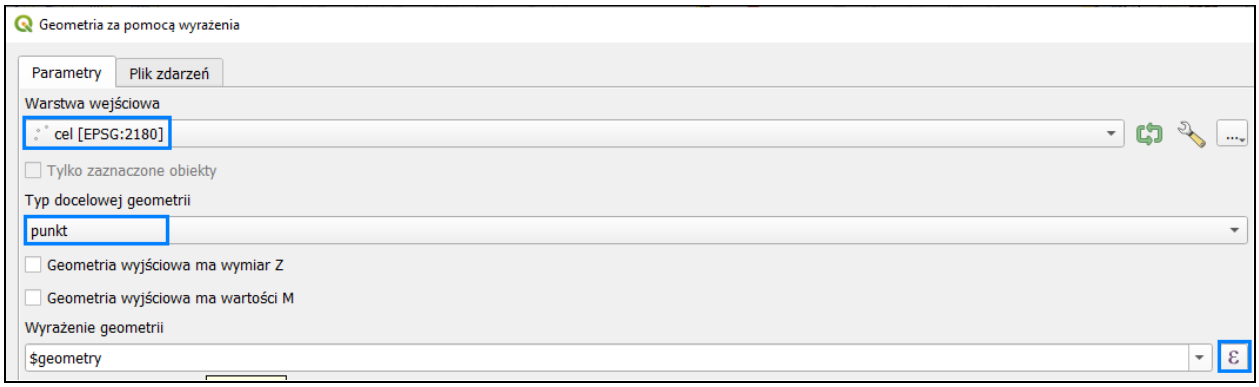
Klikamy na *Dodaj*, aby utworzyć warstwę. Jeśli formuła została przygotowana poprawnie, w oknie mapy pojawi się obiekt punktowy:



Geometrię punktową możemy wykorzystać do wygenerowania trzech punktów startowych. W tym przypadku posłużymy się wspomnianym wcześniej algorytmem *Geometria za pomocą wyrażenia*, który możemy wywołać z panelu *Algorytmów Processingu*:



W oknie dialogowym wybieramy warstwę źródłową, typ geometrii, a następnie przechodzimy do widoku kreatora wyrażeń - wywołujemy go, klikając na ikonkę po prawej stronie:

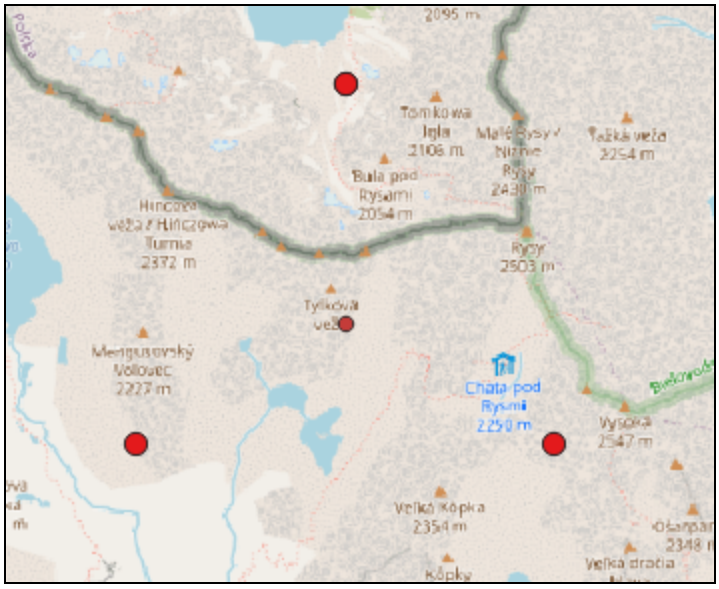


Usuwamy domyślną frazę *$geometry* i przechodzimy do stworzenia wyrażenia. Chcemy wygenerować bufor o promieniu 950 m wokół punktu docelowego, by w dalszej kolejności na jego obwodzie utworzyć trzy równomiernie rozmieszczone punkty startowe. Formuła wygląda następująco:

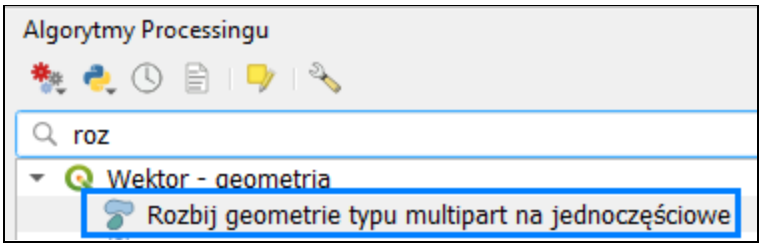
collect_geometries(
array_foreach(
array:=generate_series(120,360,120),
expression:=project($geometry, distance:=950,
azimuth:=radians(@element))))

Korzystamy z funkcji ***array_foreach()***, która działa podobnie do pętli FOR: wykonuje operację określoną w wyrażeniu na wszystkich obiektach z listy, którą użytkownik definiuje jako pierwszy argument funkcji. W tym konkretnym wyrażeniu tworzymy listę przy pomocy funkcji ***generate_series()***. Przyjmuje ona trzy argumenty: wartość startową, wartość końcową oraz interwał. Tutaj wartości te odpowiadają stopniom. Następnie wprowadzamy funkcję ***project()***. Służy ona do generowania geometrii punktowych w określonej przez użytkownika odległości (argument *distance* - tu 950 m) i odchylonych o wskazany azymut (*azimuth*: tutaj ze względu na wymóg funkcji przeliczony ze stopni na radiany i obliczony na podstawie wartości z listy ***generate_series()***. Dodatkowo całość musimy spiąć funkcją ***collect_geometries()***. Bez niej bowiem nie moglibyśmy wygenerować geometrii na podstawie listy wynikowej.

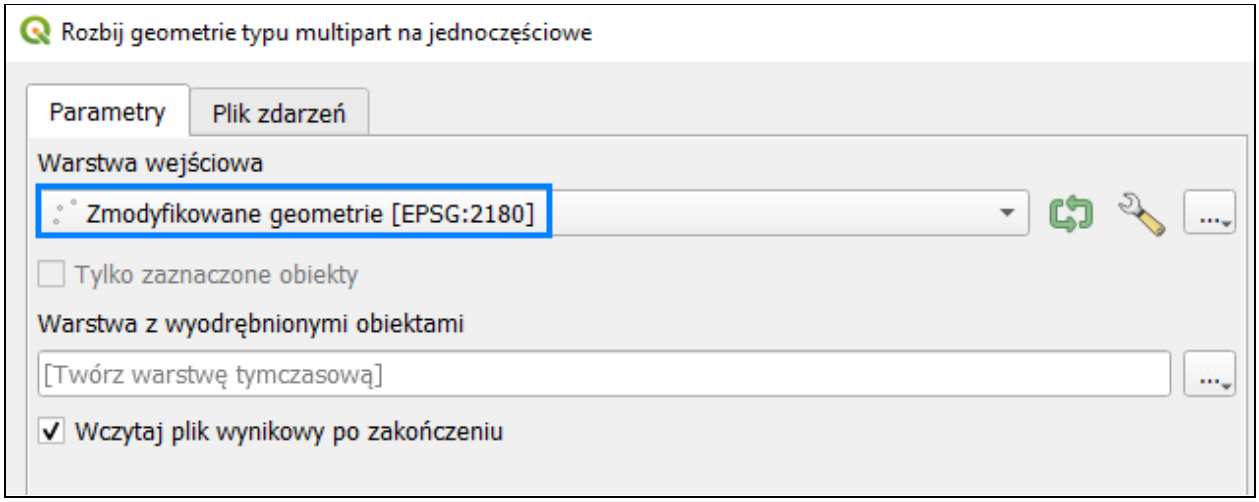
Po wprowadzeniu formuły klikamy na *OK* i uruchamiamy algorytm. Wynik działania powinien przedstawiać się następująco:



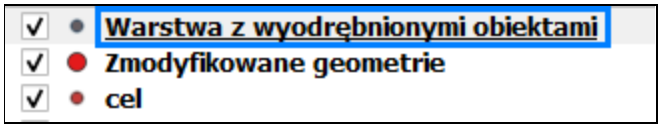
Od strony wizualnej warstwa wygląda poprawnie. Problem w tym, że zawiera jedną multigeometrię. Nam zależy na trzech odrębnych geometriach, musimy więc skorzystać z algorytmu:



Warstwą wejściową będą wygenerowane niedawno *Zmodyfikowane geometrie*:

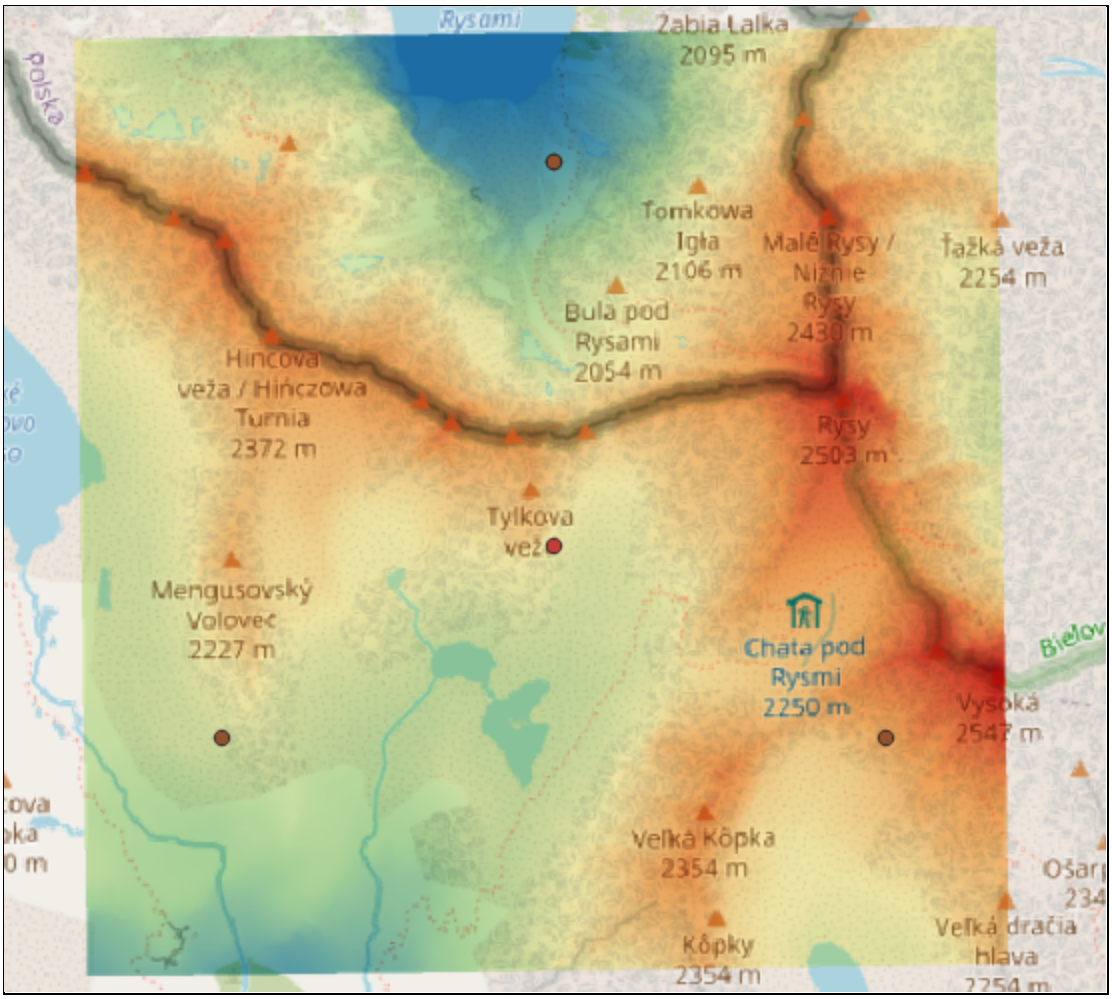


Po uruchomieniu algorytmu na liście warstw pojawi się nowa pozycja o nazwie:

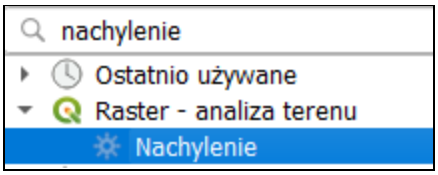


Dla porządku usuwamy poprzednią warstwę *Zmodyfikowane geometrie*. Nazwę nowoutworzonej warstwy z wyodrębnionymi obiektami zmieniamy na *start_pkt*.

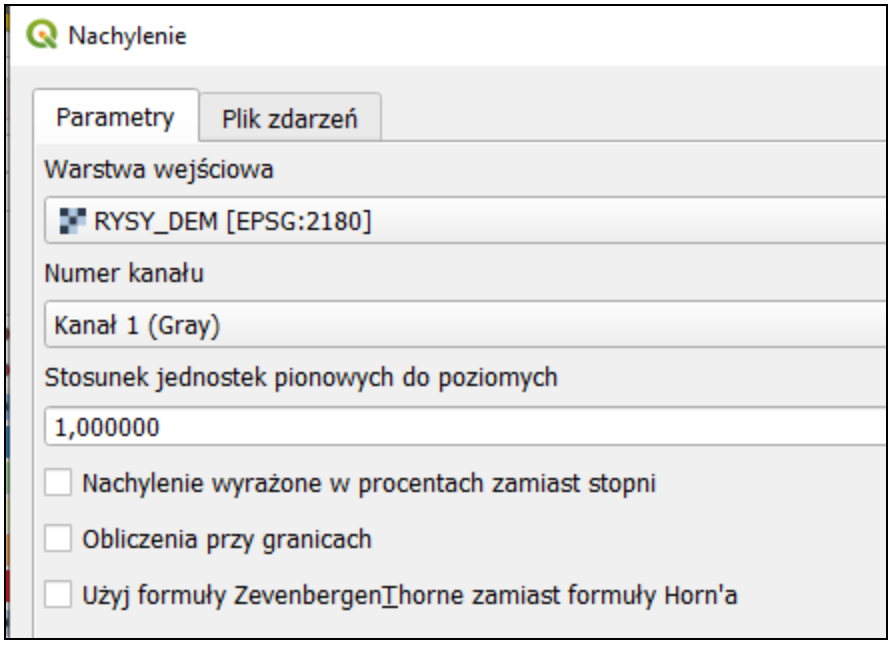
Dodajemy do projektu warstwę rastrową RYSY_DEM:



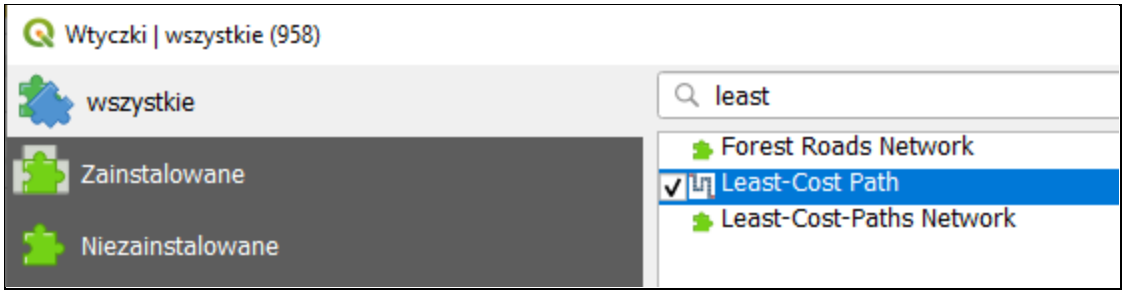
Mamy już wszystkie potrzebne obiekty punktowe, brakuje nam jednak warstwy będącej podstawą dla wygenerowania kosztów przemieszczania się i obliczenia najkrótszych tras wiodących z punktów startowych do miejsca docelowego. Dla uproszczenia przyjmiemy, że będzie to mapa spadków. Należy ją teraz utworzyć korzystając z algorytmu *Nachylenie*:



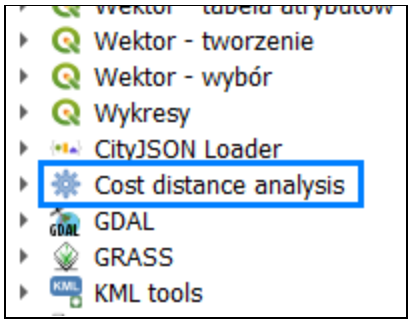
Warstwą wejściową jest oczywiście raster z Numerycznym Modelem Terenu (RYSY_DEM). Pozostałe parametry mają przydzielone wartości domyślne, których nie musimy modyfikować. Klikamy na *Uruchom* i zamykamy okno algorytmu.



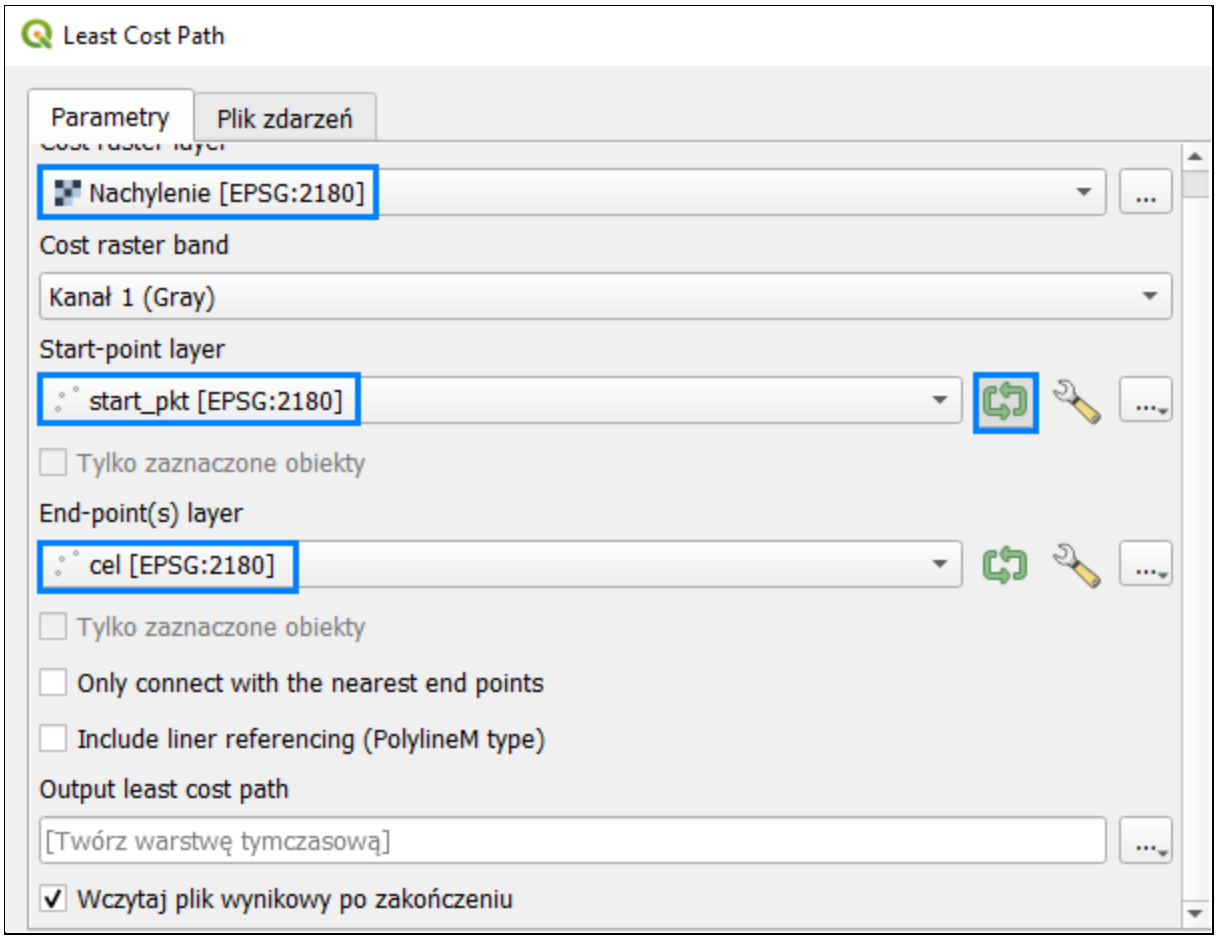
Przechodzimy teraz do instalacji wtyczki *Least Cost Path*. Plugin znajduje się w oficjalnym repozytorium, do którego dostęp uzyskujemy z poziomu *Zarządzania wtyczkami:*



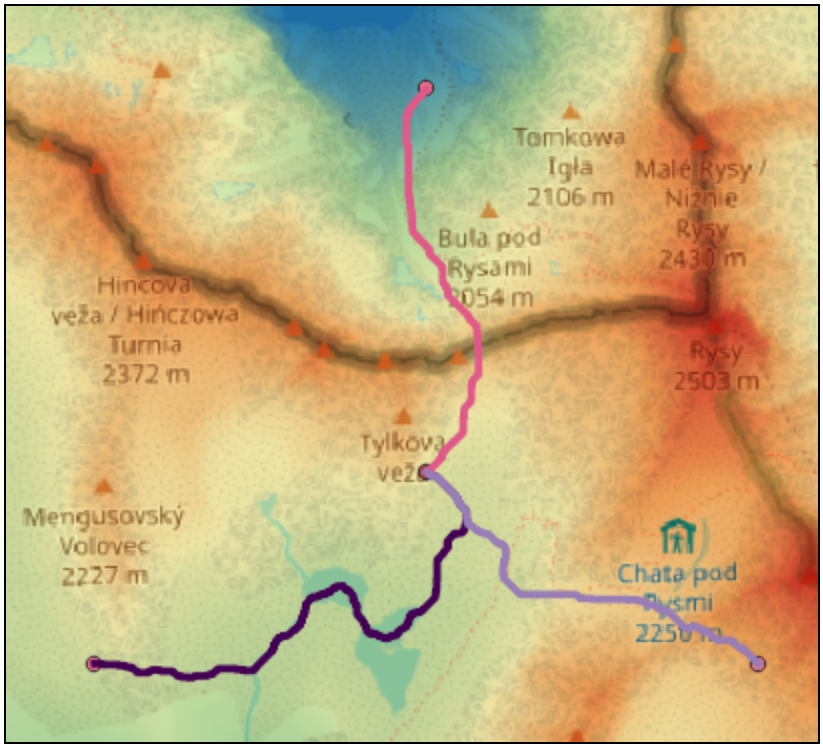
Po instalacji drzewko opcji wtyczki pojawia się w oknie panelu *Algorytmów Processingu*:



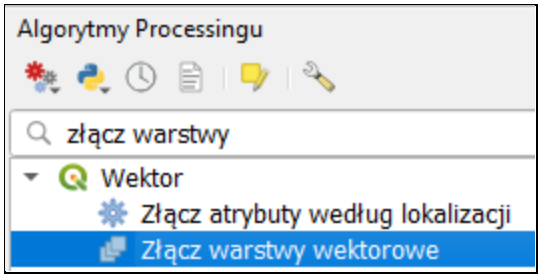
Co prawda “w środku” nie ma zbyt wielkiego wyboru, gdyż zawiera ono tylko jedną opcję o nazwie *Least Cost Path*. Uruchamiamy narzędzie, klikając na nim dwukrotnie lewym przyciskiem myszy. Przechodzimy do ustawienia parametrów przetwarzania:



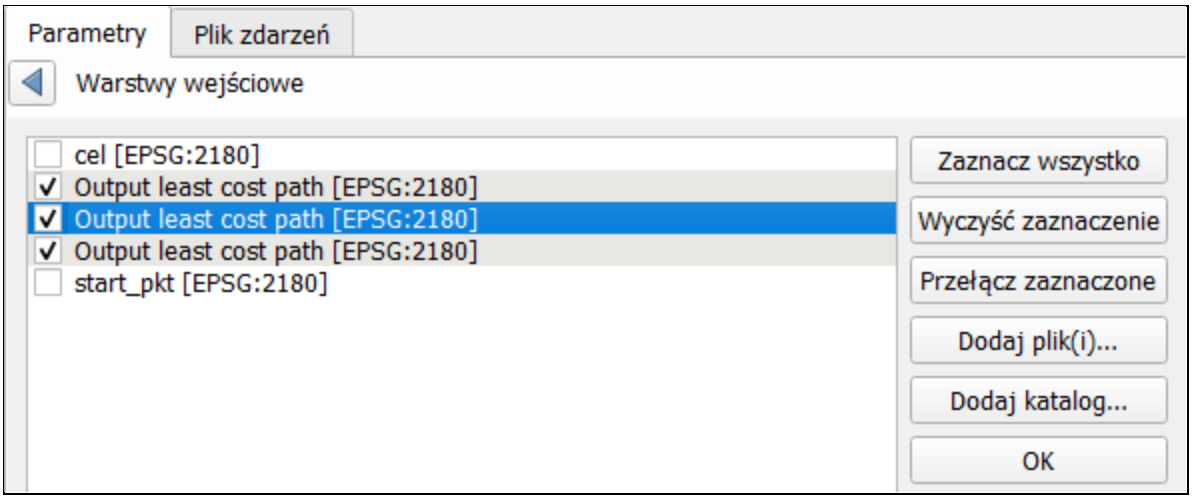
W polu *Cost raster layer* wybieramy warstwę *Nachylenie*. Punkty startowe do oczywiście warstwa *start_pkt*, punkt końcowy zaś przechowywany jest w warstwie *cel*. Dodatkowo przy polu *Start point layer(s)* zaznaczamy zieloną strzałkę, dzięki czemu program wygeneruje najkrótszą ścieżkę dla każdego obiektu z warstwy *start_pkt*. Klikamy na *Uruchom*, a po zakończeniu przetwarzania zamykamy okno i wracamy do widoku mapy.



W kolejnej części zadania spróbujemy obliczyć długości poszczególnych tras z uwzględnieniem krzywizny terenu oraz sumę podejść i zejść. Zanim jednak do tego przejdziemy, warto połączyć poszczególne trasy w jedną warstwę i nadać im numery porządkowe. Przyjmijmy, że trasa północna będzie miała nr 1, południowo-wschodnia 2, południowo-zachodnia zaś - 3. Linie łączymy za pomocą algorytmu *Złącz warstwy wektorowe*. Możemy go wywołać z poziomu panelu *Algorytmów Processingu*:

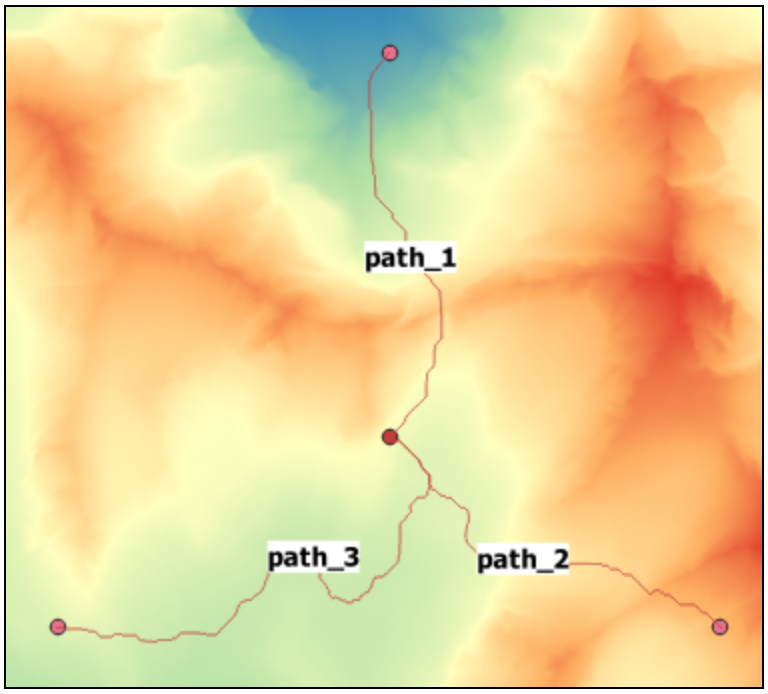


W oknie algorytmu należy wskazać warstwy do złączenia. Będą to pozycje o nazwie *Output least cost path*:



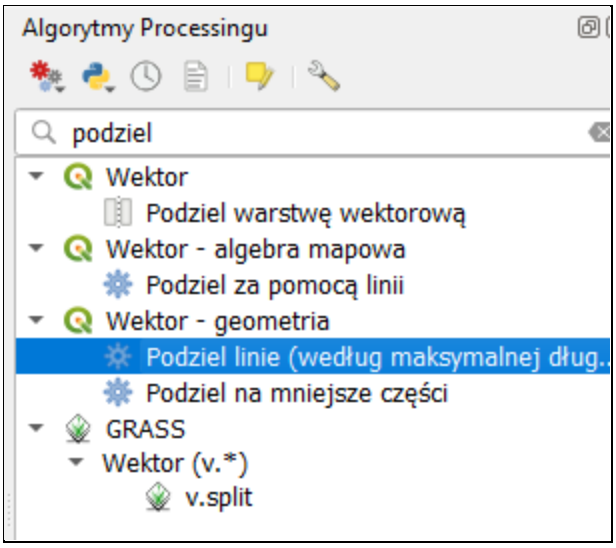
Jako docelowy układ współrzędnych wybieramy ten o kodzie EPSG:2180. Uruchamiamy algorytm, po zakończeniu jego pracy zamykamy okno i przechodzimy do widoku mapy.

Otwieramy tabelę atrybutów i modyfikujemy zawartość kolumny *layer*, nadając obiektom nazwy zgodne z przyjętą wyżej numeracją:

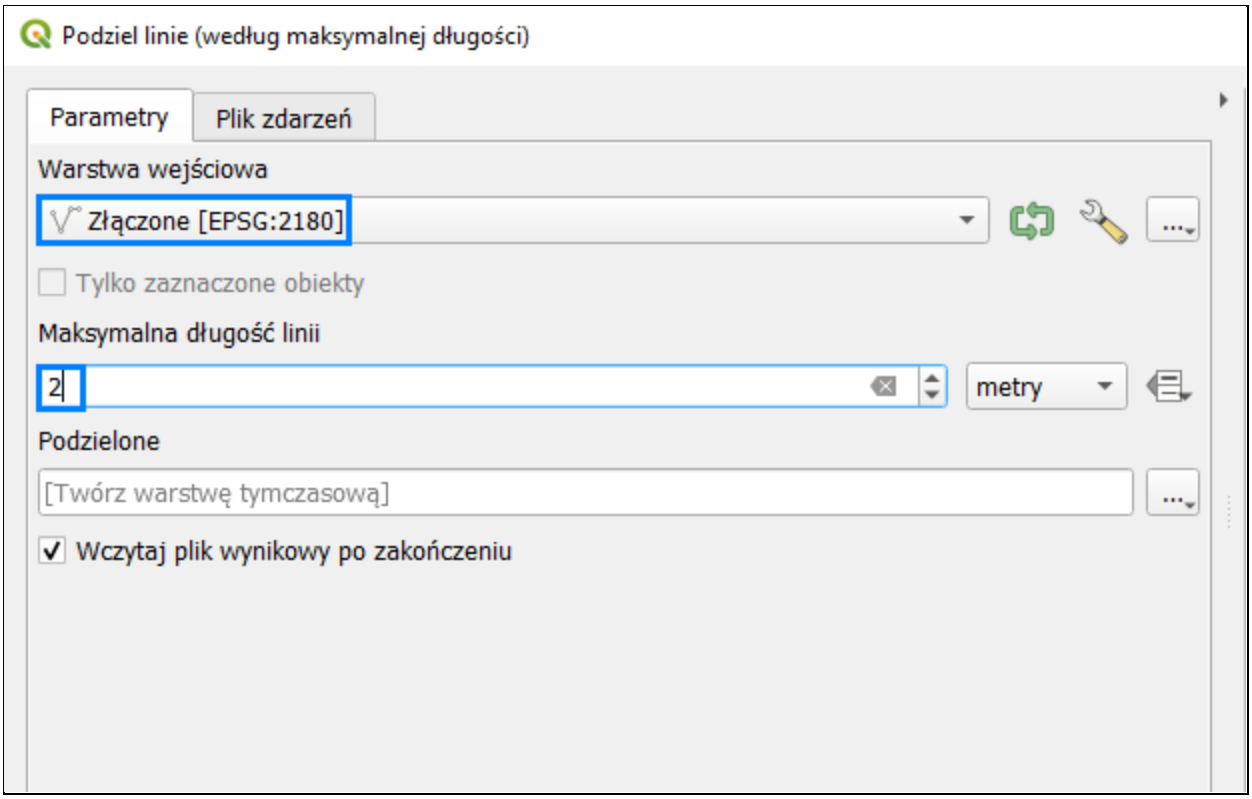


Aby obliczyć długość obiektu liniowego z uwzględnieniem ukształtowania terenu, musimy znać wysokość punktu początkowego i końcowego każdego segmentu. Dokładność pomiaru jest wprost proporcjonalna do liczby segmentów. W sytuacji, gdy obiekt liniowy posiada niewielką liczbę segmentów, konieczne jest przeprowadzenie dodatkowego podziału z wykorzystaniem

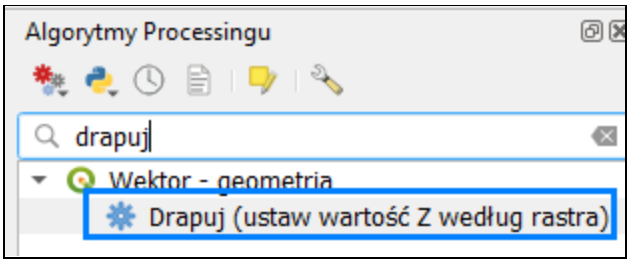
narzędzia *Podziel linie (według maksymalnej długości)*. Algorytm znajdziemy w panelu *Algorytmów Processingu*:



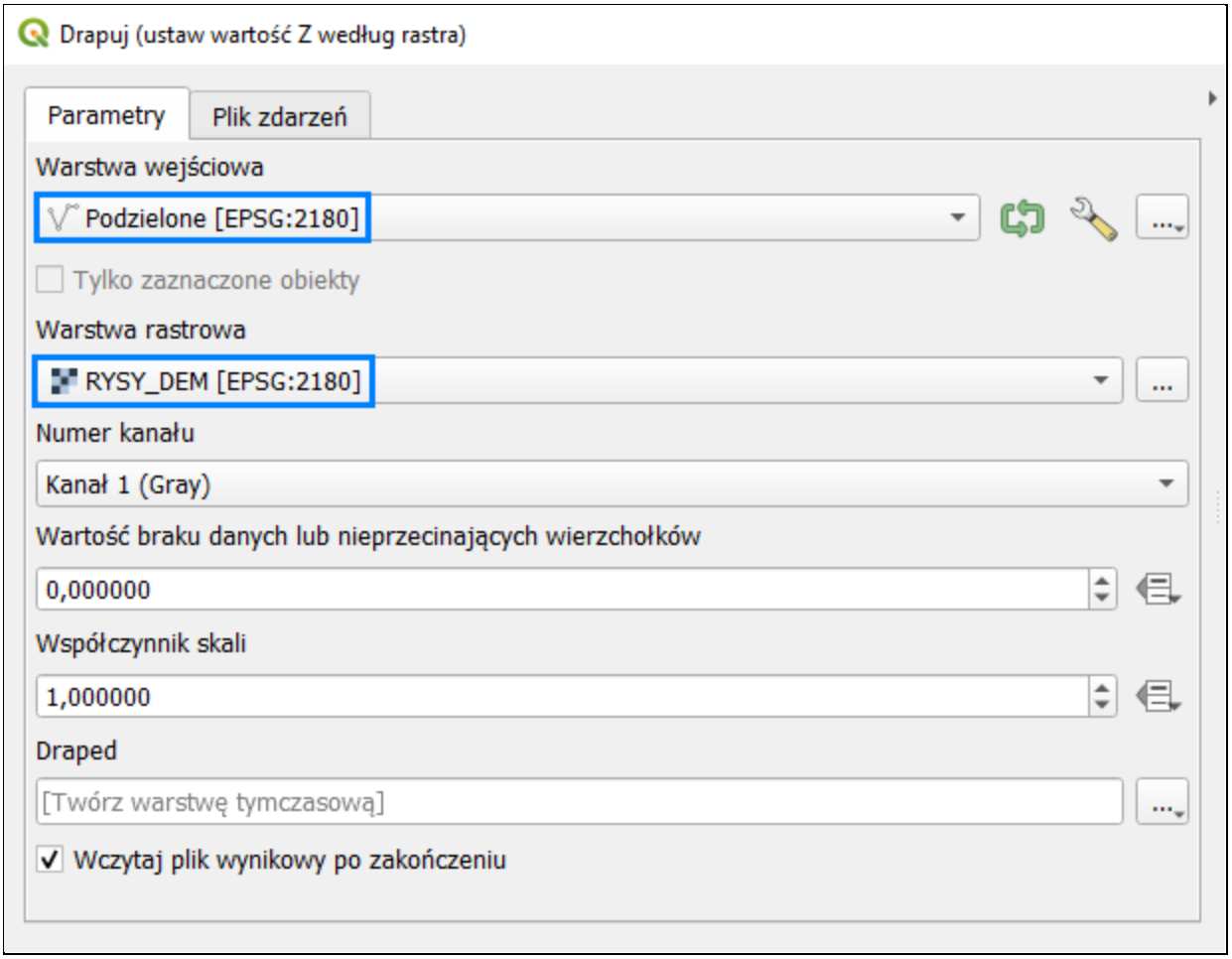
Uruchamiamy narzędzie. Jako warstwę wejściową wskazujemy *Złączone*. Maksymalną długość linii ustalamy na dwa metry. Zapewni nam to relatywnie wysoką dokładność pomiaru.



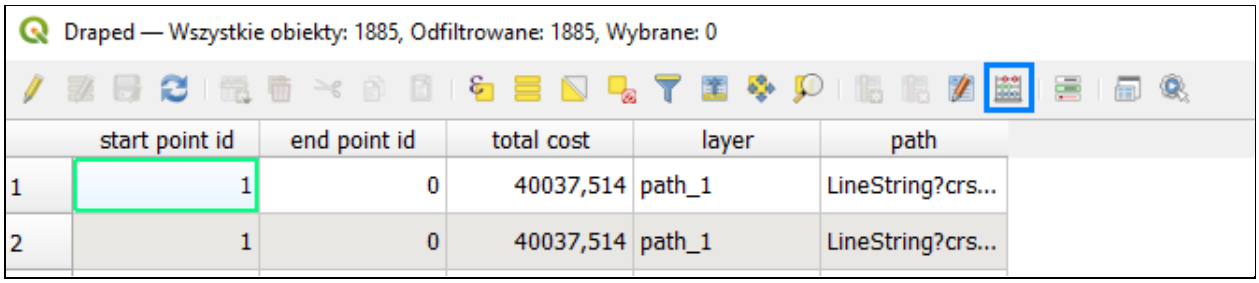
Po zakończeniu działania zamykamy okno dialogowe i przechodzimy do okna mapy. Teraz możemy przypisać wierzchołkom poszczególnych segmentów wartość Z niezbędną do przeprowadzenia dalszych obliczeń. Wykorzystamy do tego narzędzie *Drapuj (...)*:



Drapowanie działa podobnie do próbkowania wartości rastra za pomocą warstwy punktowej; każdemu wierzchołkowi segmentu linii przypisywana jest wartość Z odczytywana z rastra wskazanego przez użytkownika:



Po wykonaniu działania na liście warstw pojawi się nowa pozycja o nazwie *Draped*. Przejdźmy teraz do jej tabeli atrybutów i otwórzmy *Kalkulator Pól*:



	start point id	end point id	total cost	layer	path
1	1	0	40037,514	path_1	LineString?crs...
2	1	0	40037,514	path_1	LineString?crs...

Długość linii 3D obliczamy na podstawie poniższego wzoru:

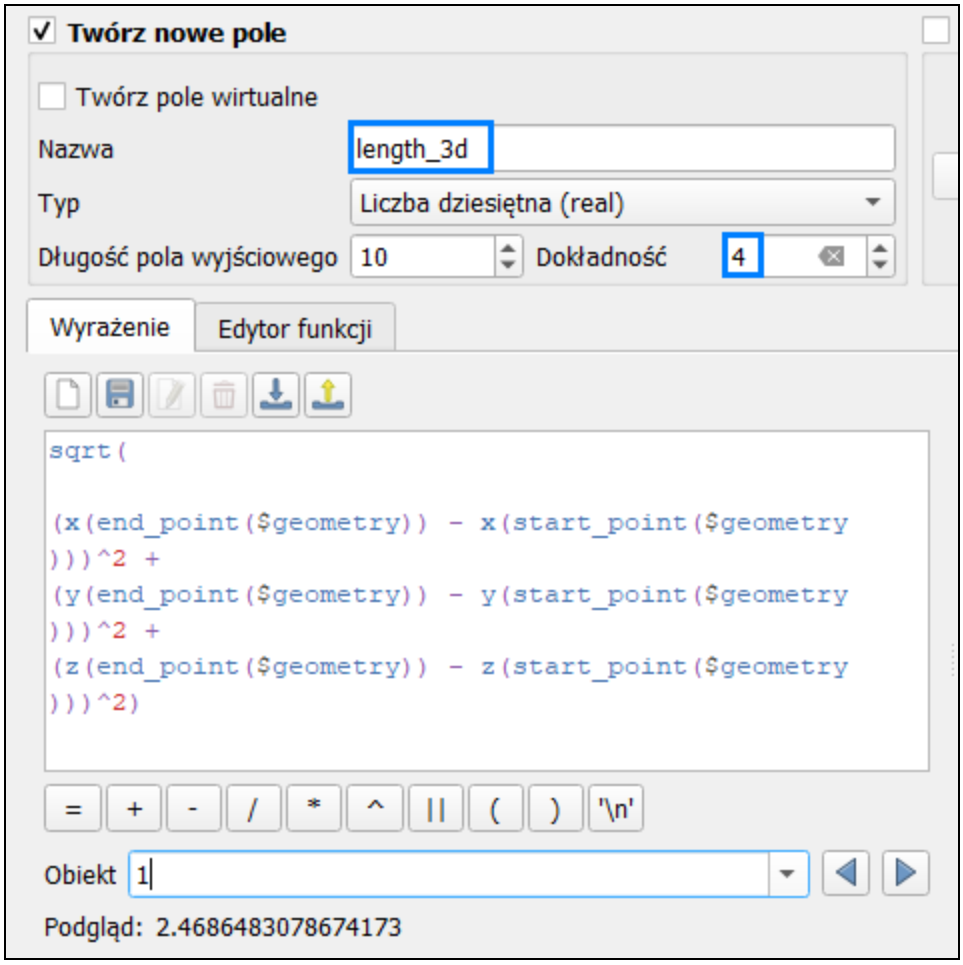
$$d = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2}$$

Jest to pierwiastek kwadratowy z sumy różnic kwadratów wartości x,y i z. W wydaniu QGISowym będzie to wyglądało tak:

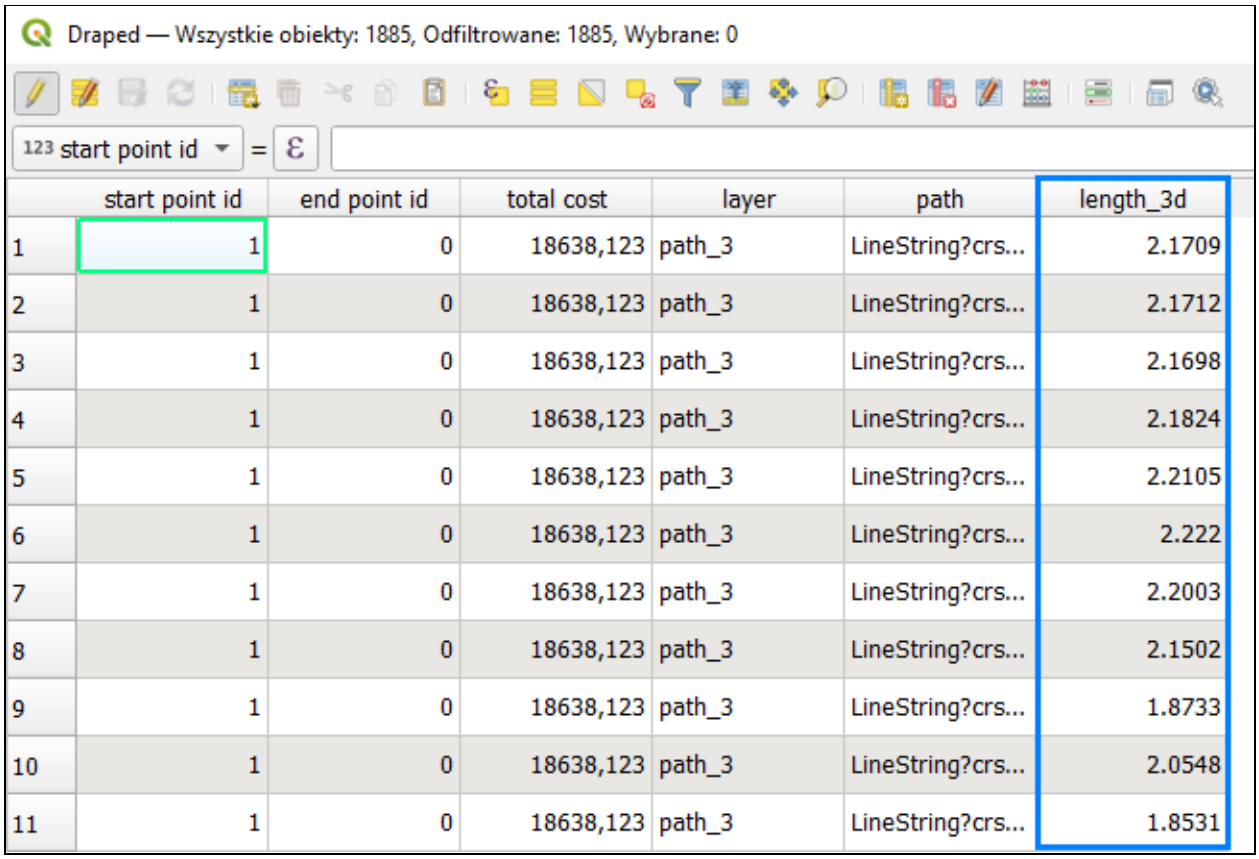
sqrt(

(x(end_point($geometry)) - x(start_point($geometry)))^2 +
(y(end_point($geometry)) - y(start_point($geometry)))^2 +
(z(end_point($geometry)) - z(start_point($geometry)))^2)

Po wprowadzeniu formuły powinien wyświetlić się podgląd wyniku w postaci liczby całkowitej o wartości oscylującej w okolicach 2m:



Wyniki obliczeń zapisujemy w nowej kolumnie o nazwie *length_3d*. Typ nowego pola - *liczba dziesiętna* o precyzji 4. Klikamy na *OK*. Po przetworzeniu danych wracamy do okna tabeli atrybutów i sprawdzamy, czy dane w nowej kolumnie wyświetlają się poprawnie:



	start point id	end point id	total cost	layer	path	length_3d
1	1	0	18638,123	path_3	LineString?crs...	2.1709
2	1	0	18638,123	path_3	LineString?crs...	2.1712
3	1	0	18638,123	path_3	LineString?crs...	2.1698
4	1	0	18638,123	path_3	LineString?crs...	2.1824
5	1	0	18638,123	path_3	LineString?crs...	2.2105
6	1	0	18638,123	path_3	LineString?crs...	2.222
7	1	0	18638,123	path_3	LineString?crs...	2.2003
8	1	0	18638,123	path_3	LineString?crs...	2.1502
9	1	0	18638,123	path_3	LineString?crs...	1.8733
10	1	0	18638,123	path_3	LineString?crs...	2.0548
11	1	0	18638,123	path_3	LineString?crs...	1.8531

Zależy nam na poznaniu sumy poszczególnych długości dla wszystkich trzech dróg. Stworzymy więc nową tabelę wynikową bez geometrii, zawierającą odpowiednie wyliczenia. Do tego celu wykorzystamy *warstwę wirtualną*. Okno kreatora takiej warstwy możemy otworzyć "na skróty", klikając na ikonkę

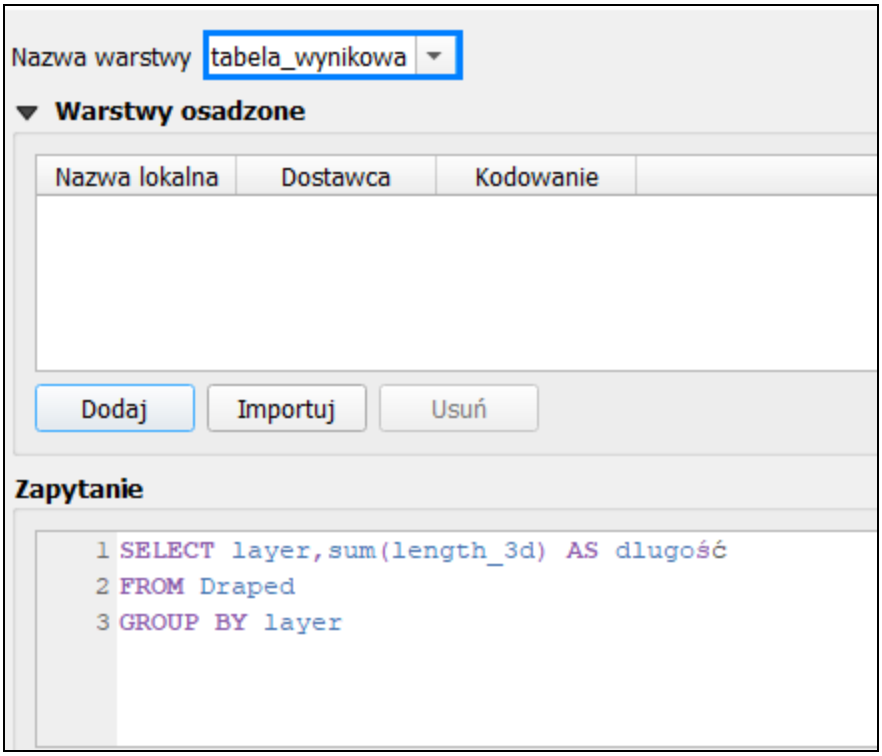
 , znajdującą się na pasku narzędzi QGIS.

Przejdźmy do przygotowania formuły. Celem jest uzyskanie tabeli z dwiema kolumnami, zawierającymi odpowiednio nazwę trasy oraz łączną długość liczoną z uwzględnieniem ukształtowania terenu. Liczba rekordów powinna być zgodna z liczbą unikalnych wartości w polu *layer*.

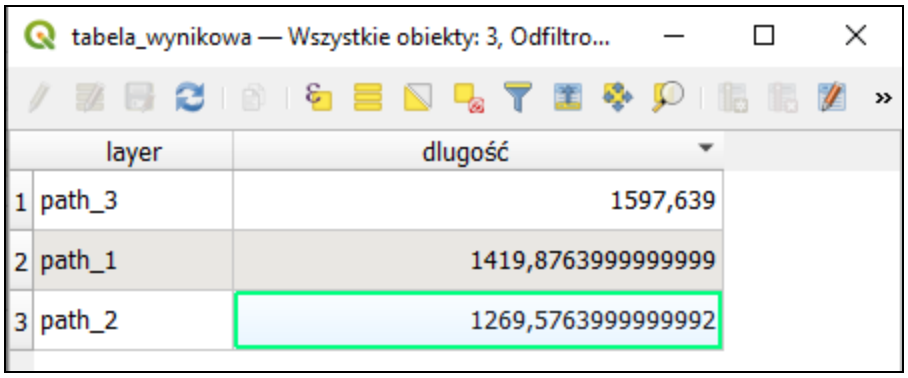
Przykładowe wyrażenie wygląda następująco:

SELECT layer,sum(length_3d) AS długość
FROM Draped
GROUP BY layer

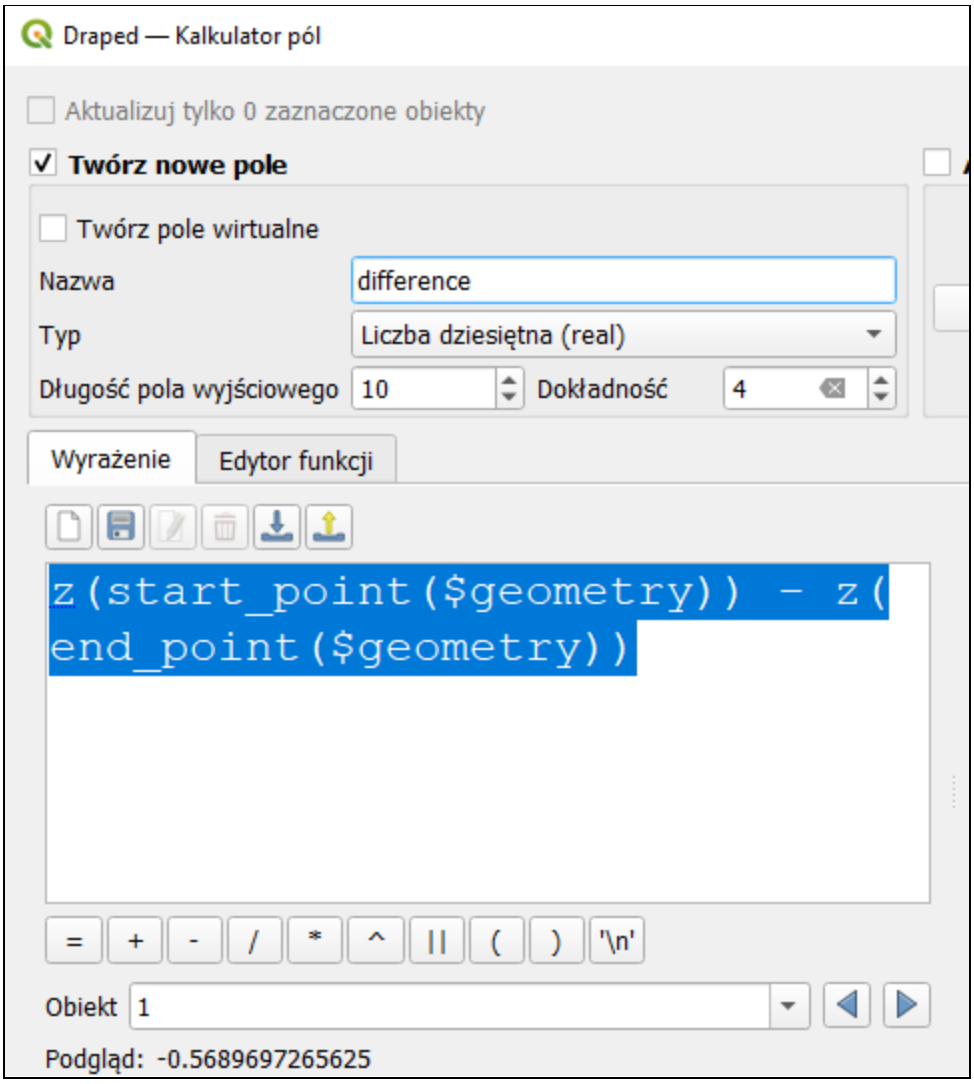
Pamiętajmy również o zmianie nazwy warstwy wirtualnej, np. na *tabela_wynikowa*:



Po kliknięciu na *Dodaj* warstwa z tabelą powinna pojawić się na liście w panelu warstw. Nie zawiera ona geometrii, niemniej mamy swobodny dostęp do zawartości tabeli atrybutów:

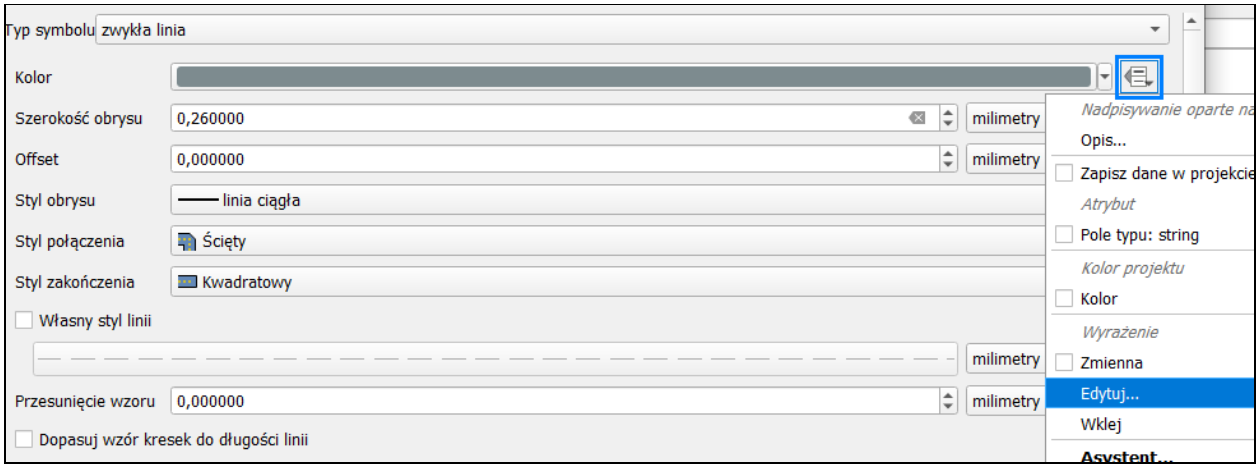


Z zestawienia wynika, że najwięcej do przejścia ma osoba startująca z lokacji południowo-zachodniej. Spróbujmy teraz obliczyć sumę zejść i podejść dla poszczególnych tras. Obliczenia możemy przeprowadzić w ramach warstwy *Draped*. Przejdźmy więc do *Kalkulatora Pól* warstwy i przygotujmy odpowiednią formułę. Interesuje nas różnica wysokości między wartościami Z dla wierzchołka początkowego i końcowego:



Dane wynikowe zapisujemy w nowej kolumnie *difference*, typ *liczba dziesiętna*, precyzja 4. Sprawdźmy kontrolnie, czy dane poprawnie zapisały się w tabeli atrybutów. Jeśli tak, możemy zamknąć widok tabeli i przejść do właściwości warstwy *Draped*. Wykorzystamy teraz dane z kolumny *difference* do przygotowania dynamicznej symbolizacji rozróżniającej podejścia i zejścia.

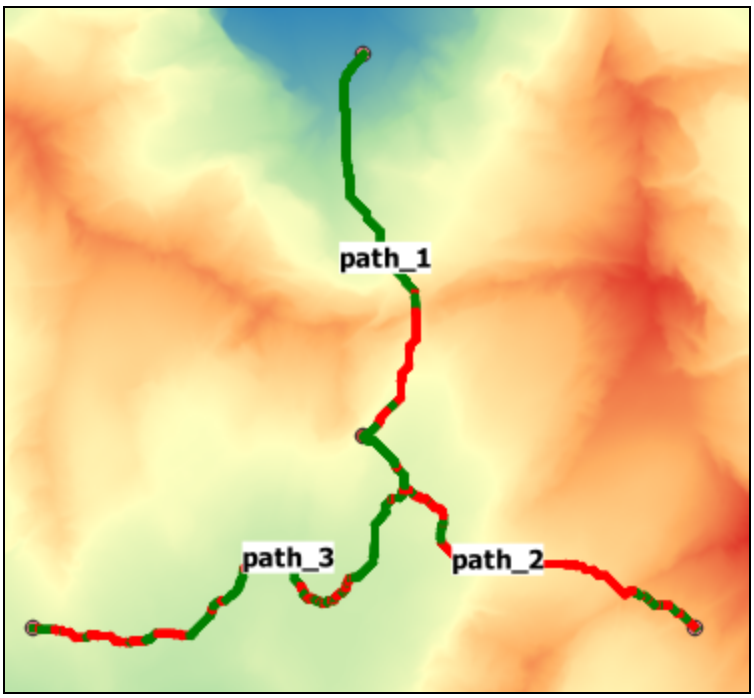
Otwieramy zakładkę *Styl* i przy polu z kolorem linii wybieramy *Nadpisywanie oparte na danych*, a następnie *Edytuj*:



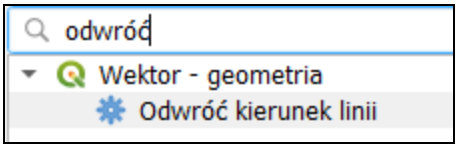
Posłużymy się prostym wyrażeniem, które zaznaczy odcinki podejść na czerwono, zejść zaś - na zielono:

if("difference" > 0, 'red','green')

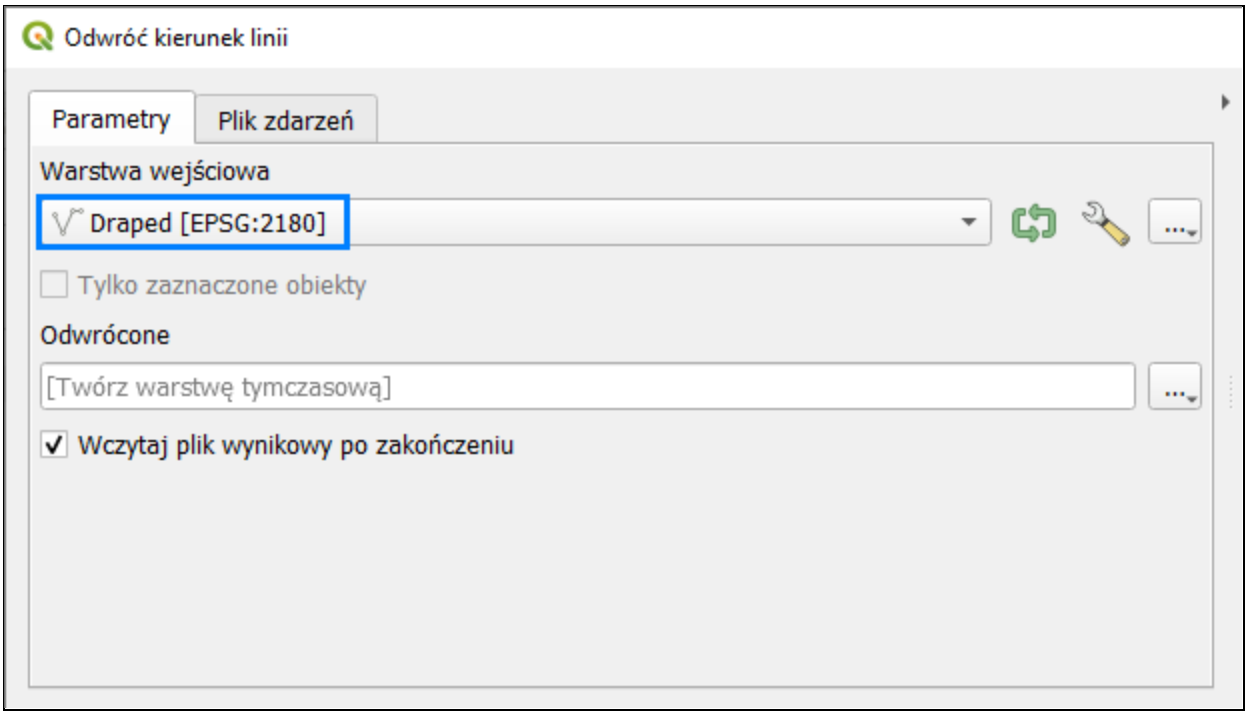
Warto również zwiększyć grubość linii do 1 mm, dzięki czemu osie tras będą lepiej widoczne. Po wprowadzeniu zmian warstwa powinna prezentować się następująco:



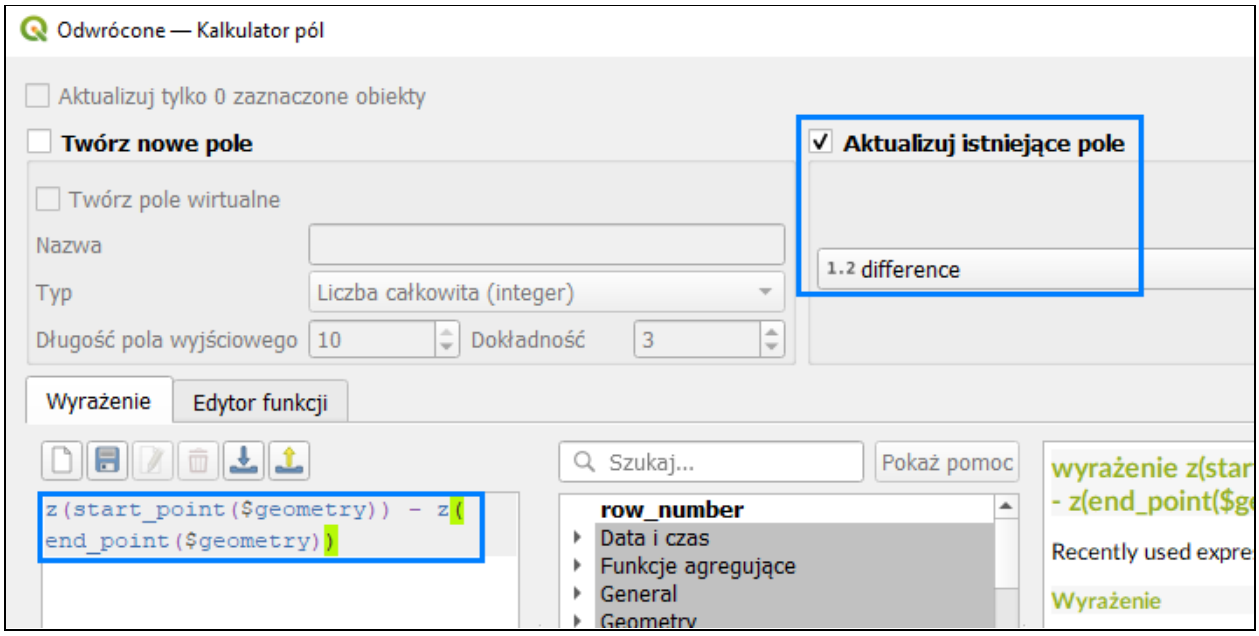
Niestety już na pierwszy rzut oka widać, że coś jest nie tak - odcinki “pod górę” zostały oznaczone kolorem zielonym, zarezerwowanym dla zejść. Aby rozwiązać problem odwróćmy kierunek linii, korzystając z odpowiedniego algorytmu:



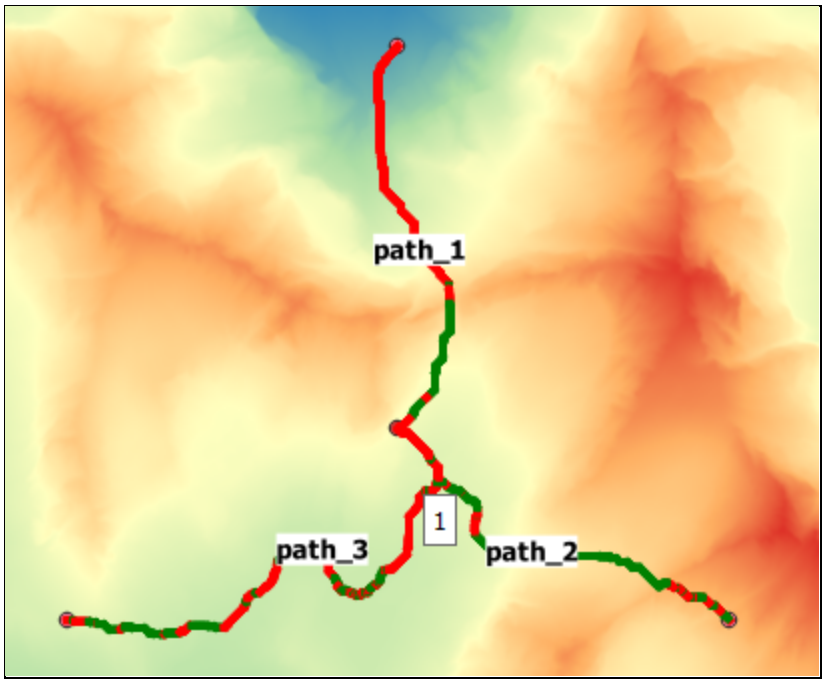
Kierunek linii odwracamy oczywiście dla warstwy *Draped*:



W efekcie uzyskamy nową warstwę o nazwie *Odwrócone*. Otwieramy jej tabelę atrybutów i ponownie przeliczamy wartości w polu *difference*, korzystając z wcześniejszej formuły:



Po przeliczeniu zapisujemy zmiany w warstwie i przechodzimy do jej symbolizacji. Nadajemy jej ten sam dynamiczny styl, wykorzystujący *Nadpisywanie oparte na danych*. Efekt powinien być jednak zgoła odwrotny:



Skoro wizualizacja podejść i zejść koresponduje już z rzeźbą terenu, możemy przejść do wielkiego finału, tj. obliczenia sumy zejść i podejść dla wszystkich trzech szlaków. Przygotowanie tabeli wynikowej tylko przy użyciu narzędzi QGIS byłoby bardzo trudne (jeśli nie

niemożliwe), w związku z czym ponownie skorzystamy z warstw wirtualnych. Zanim jednak do tego przejdziemy, zaprowadźmy porządek w naszych warstwach. Usuńmy warstwę *Draped*, natomiast warstwę *Odwrócone* przemianujmy na *szlaki*.

Utwórzmy teraz nową warstwę wirtualną i w oknie kreatora umieśćmy następującą formułę:

SELECT a.layer, b. wejscia, c.zejscia
FROM szlaki a
JOIN (SELECT layer, sum(difference) AS wejscia -- "wyciągamy" sumę wartości z kolumny difference
FROM szlaki
WHERE difference > 0 -- eliminujemy z zestawienia rekordy reprezentujące zejścia
GROUP BY layer)b -- tworzymy tabelę z sumą wejść i przyłączamy wyniki do danych źródłowych
ON a.layer = b.layer
JOIN (SELECT layer, sum(difference) AS zejscia -- wyciągamy sumę wartości z kolumny difference
FROM szlaki
WHERE difference < 0 -- eliminujemy z zestawienia rekordy reprezentujące wejścia
GROUP BY layer)c -- tworzymy tabelę z sumą zejść i przyłączamy wyniki do danych źródłowych
ON a.layer = c.layer
GROUP BY a.layer -- grupujemy dane na podstawie nazw szlaków, dzięki czemu otrzymujemy trzy rekordy wynikowe w miejsce kilku tysięcy

UWAGA! Zapis następujący po znaku "--" stanowi komentarz do kodu i nie jest brany pod uwagę w procesie przetwarzania zapytania. Wskazówki mają na celu lepsze zrozumienie działania poszczególnych elementów instrukcji sql. Można skopiować cały zapis i uruchomić kwerendę w oknie kreatora warstw wirtualnych.

Nazwijmy warstwę wynikową *suma_wejsc_i_zejsc*. Po wprowadzeniu formuły klikamy na *Dodaj*. Otwieramy tabelę atrybutów warstwy wynikowej i odczytujemy wartości.

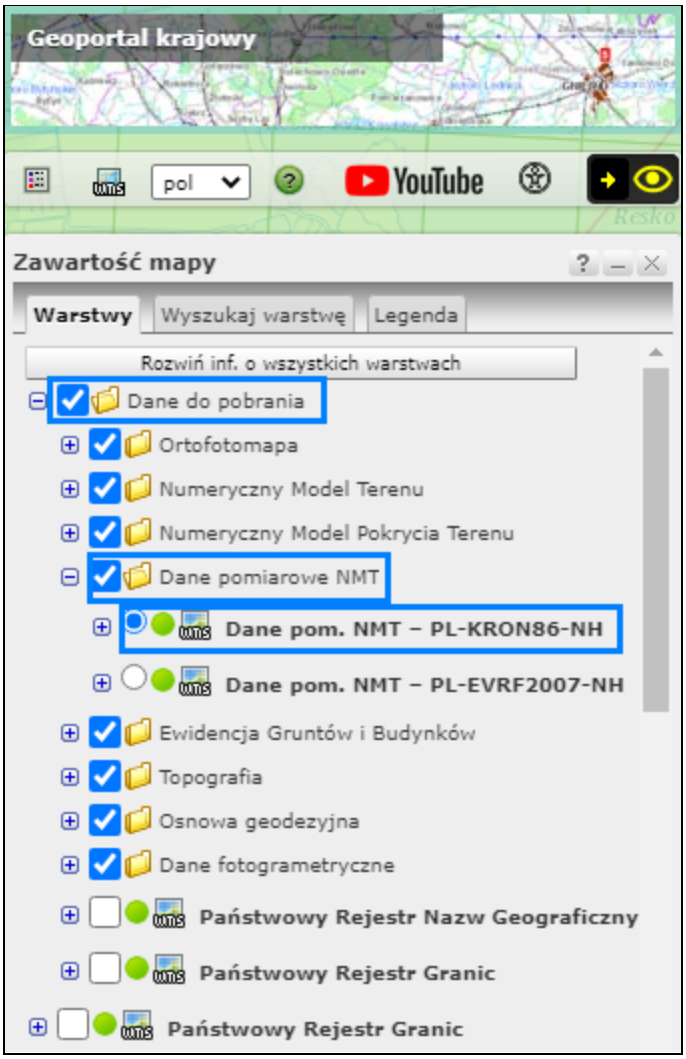
Dane LIDAR w QGIS. Omówienie funkcjonalności i przykłady zastosowania wtyczki LAS Tools

Pobieranie danych

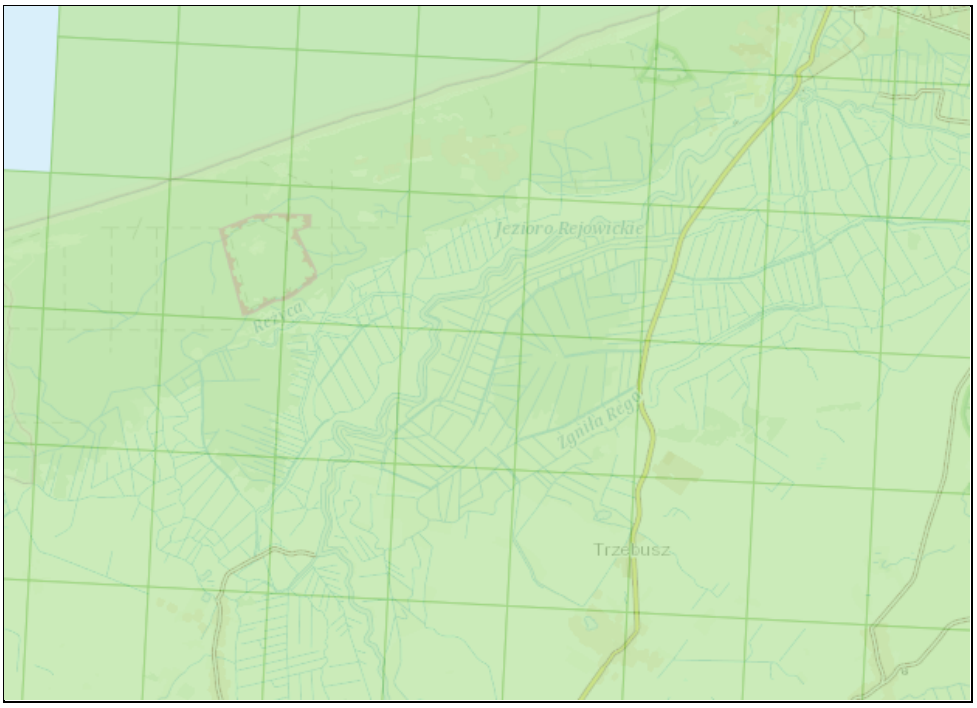
Lidar (Light Detection and Ranging) to metoda zdalnej (teledetekcyjnej) rejestracji informacji o ukształtowaniu i pokryciu terenu. Specjalna aparatura, umocowana przeważnie na spodzie statku powietrznego, emituje wiązkę lasera, która odbija się od podłoża i powraca do rejestratora; na podstawie czasu powrotu wiązki określany jest dystans między emiterem a podłożem. W efekcie uzyskuje się zapis zmierzonej lokalizacji uwzględniający współrzędne X,Y oraz Z (wysokość).

Dane Lidar w formacie .las lub .laz można pobrać nieodpłatnie z witryny *www.geoportal.gov.pl.* W oknie głównym witryny należy po prawej stronie wybrać *Geoportal krajowy*:

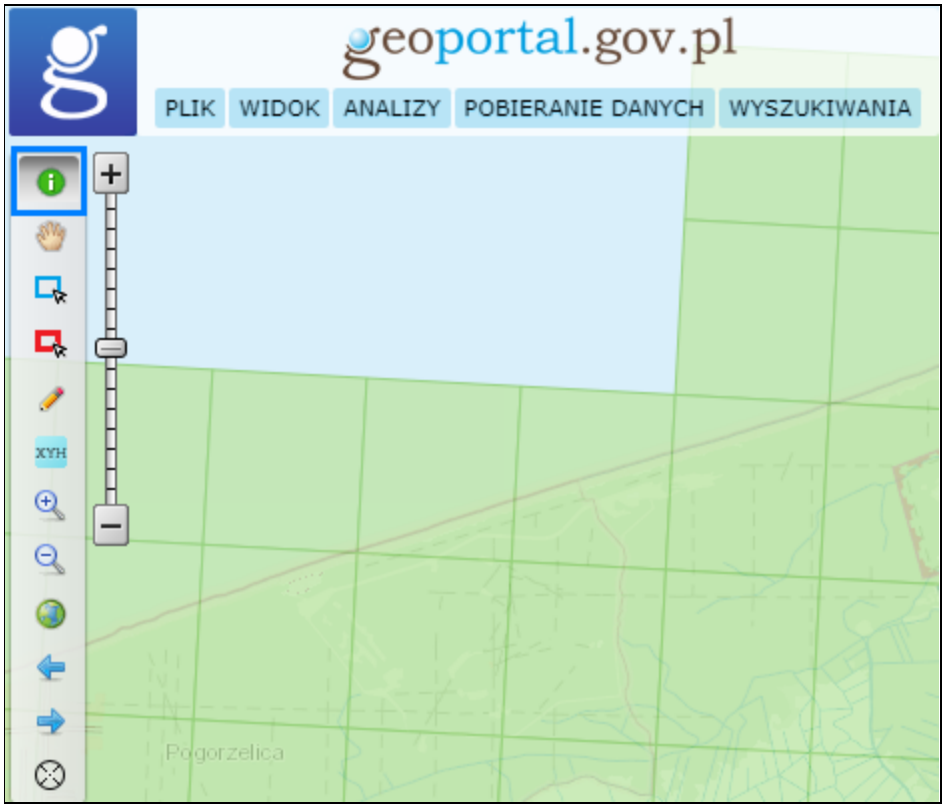
W oknie zawartości mapy, domyślnie wyświetlającym się po prawej stronie, należy wybrać odpowiednio *Dane do pobrania*, *Dane pomiarowe NMT* oraz wersję *PLKRON86* (...):



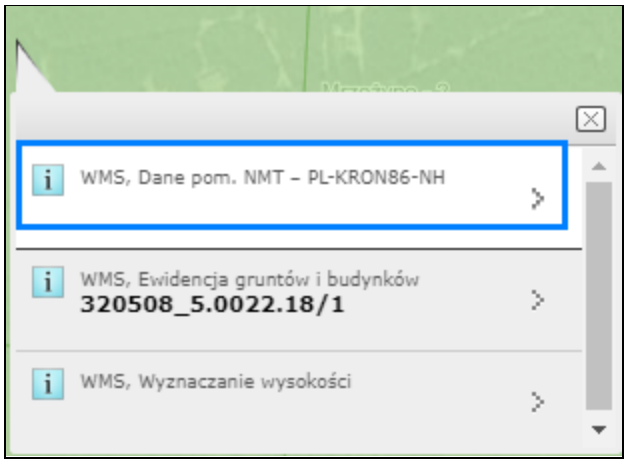
Po zaznaczeniu wskazanej opcji ekran główny zmieni wygląd, zyskując zielone, półprzezroczyste wypełnienie. Obrazuje ono zasięg wybranej usługi (pełne pokrycie dla całego kraju):



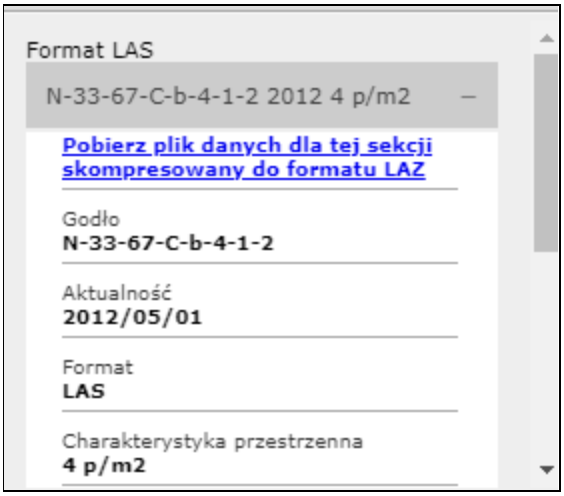
Aby pobrać dane, należy w pierwszej kolejności upewnić się, wertykalnie rozciągniętym pasku narzędzi po prawej stronie zaznaczona jest pierwsza ikona od góry:



Następnie należy kliknąć lewym przyciskiem myszy w obrębie granic wybranego oczka siatki, wyświetlającej się w głównym oknie usługi. Po kliknięciu pojawi się nowe okno, w którym rozwijamy pierwszą opcję z góry:



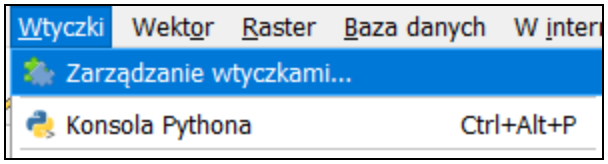
W kolejnym oknie mamy możliwość pobrania skompresowanych danych Lidar w formacie .laz. Dodatkowo możemy zapoznać się z informacjami na temat godła i aktualności wybranego opracowania, charakterystyki przestrzennej (gęstości pomiaru), etc.:



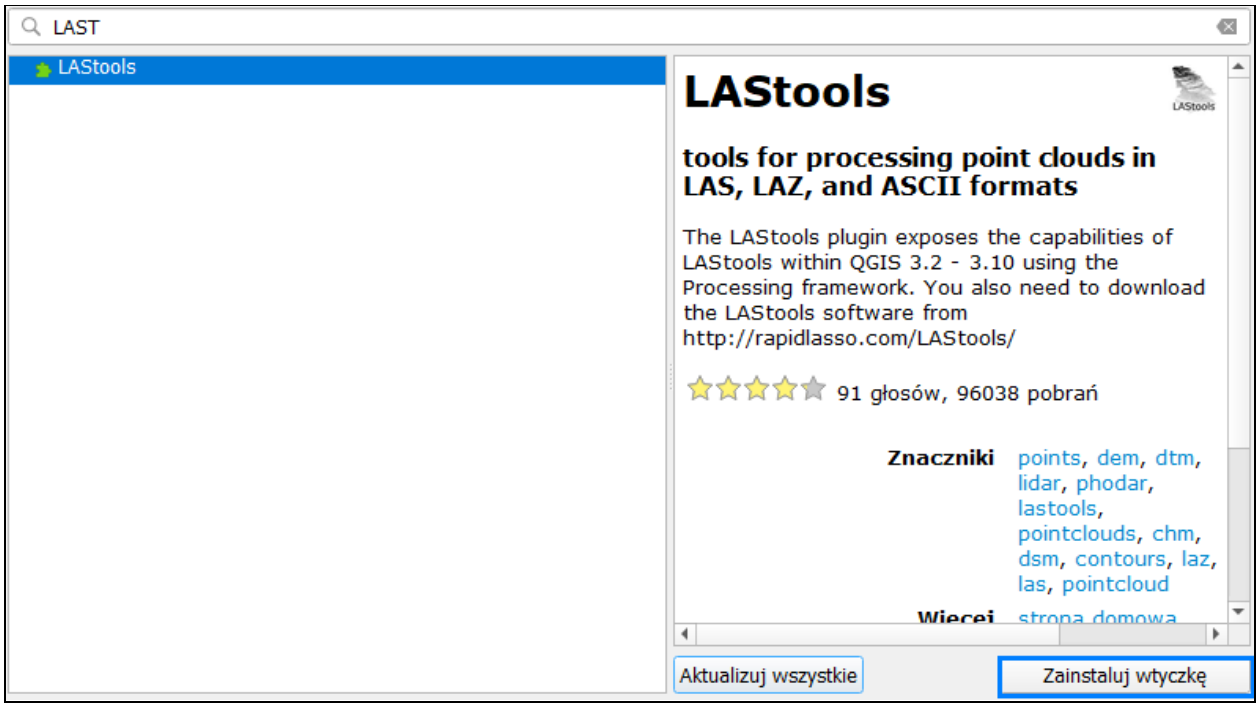
Instalacja i aktywacja narzędzia LAStools

Do obsługi danych przestrzennych w formacie .las wykorzystamy pakiet dodatkowych narzędzi o nazwie LASTools. Można go pobrać ze strony https://rapidlasso.com/lastools/ . Proces pobierania rozpocznie się po kliknięciu na *Download*:

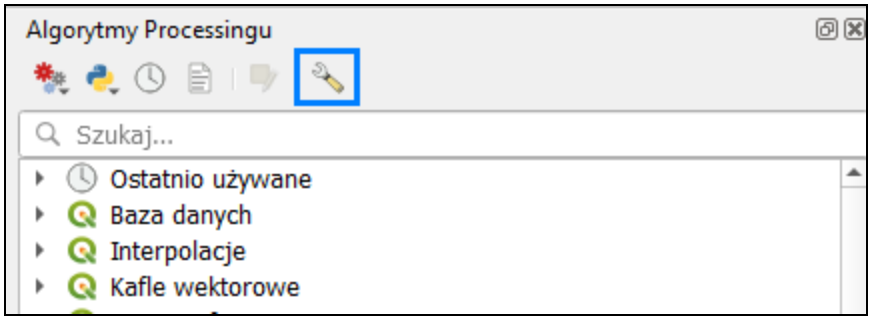
Po pobraniu rozpakowujemy archiwum i umieszczamy pliki w katalogu *C:\LAStools*. Do obsługi narzędzi pobranego pakietu niezbędna jest instalacji wtyczki LASTools w programie QGIS. Wtyczkę możemy pobrać za pośrednictwem opcji *Wtyczki ->Zarządzanie wtyczkami* - do wyboru w oknie programu:



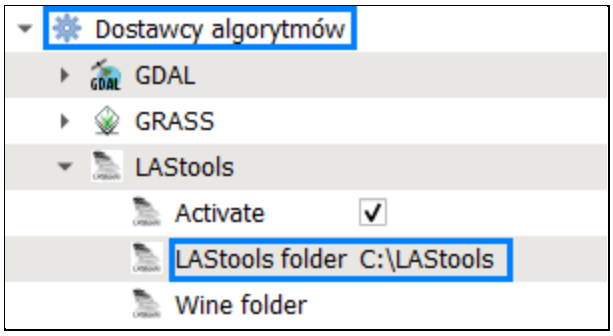
Wtyczkę umieszczono w oficjalnym repozytorium QGIS. Aby ją zlokalizować i zainstalować, należy wybrać zakładkę *Wszystkie*, wpisać nazwę wtyczki w oknie *Szukaj*, wybrać znaleziony rekord i kliknąć na *Zainstaluj wtyczkę* w prawym dolnym rogu okna:



Po zainstalowaniu wtyczki konieczne jest również zdefiniowanie ścieżki dostępu do narzędzi *LAStools*. W tym celu należy rozwinąć panel *Algorytmów Processingu* i z górnego paska paneli wybrać *Opcje*:



W kolejnym oknie rozwijamy drzewo *Dostawcy algorytmów*, wybieramy *LAStools* i w razie potrzeby ręcznie modyfikujemy ścieżkę tak, by wskazywała ona katalog C:\LAStools :



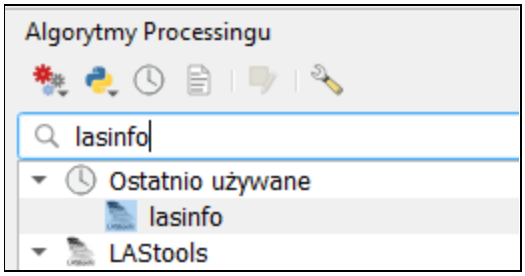
Zatwierdzamy zmiany, klikając na *OK* w dolnym prawym rogu ekranu.

Obsługa LASTools

Na wstępie należy zaznaczyć, że jest to oprogramowanie w pewnej mierze komercyjne, dlatego też część opcji nie jest dostępna. Uwaga ta dotyczy przede wszystkim możliwości przetwarzania chmur punktów o liczebności większej niż 10^6. Z drugiej strony dostawca usługi zawarł w swojej aplikacji narzędzia, które pozwalają obejść wspomniane ograniczenie.

Zanim przejdziemy do przetwarzania danych, utwórzmy katalog *C:\lidar_dane* i przenieśmy do niego wcześniej pobrany plik w formacie .las.

Następnie otwórzmy panel *Algorytmów Processingu* i w oknie *Szukaj* wpiszmy *lasinfo*.



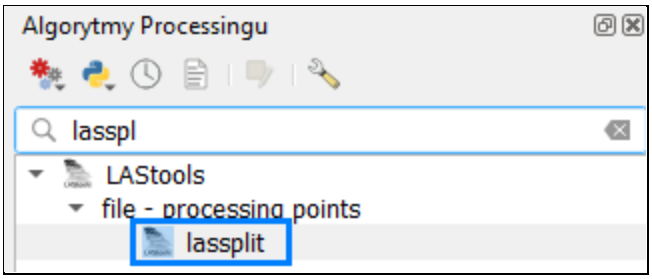
Jest to narzędzie do generowania specyfikacji pliku, uwzględniającej m.in. użyty układ współrzędnych, liczbę punktów, kody klasyfikacji, etc. Uruchamiamy algorytm i wskazujemy ścieżkę do naszego pliku .laz. Pozostałe opcje pozostawiamy bez zmian. Klikamy na *Uruchom*. Po chwili w oknie algorytmu powinno wyświetlić się zestawienie informacji:

```
histogram of classification of points:
2025 never classified (0)
1727578 ground (2)
1247048 low vegetation (3)
79991 medium vegetation (4)
3911952 high vegetation (5)
7776 building (6)
132 noise (7)
3321176 overlap (12)
```

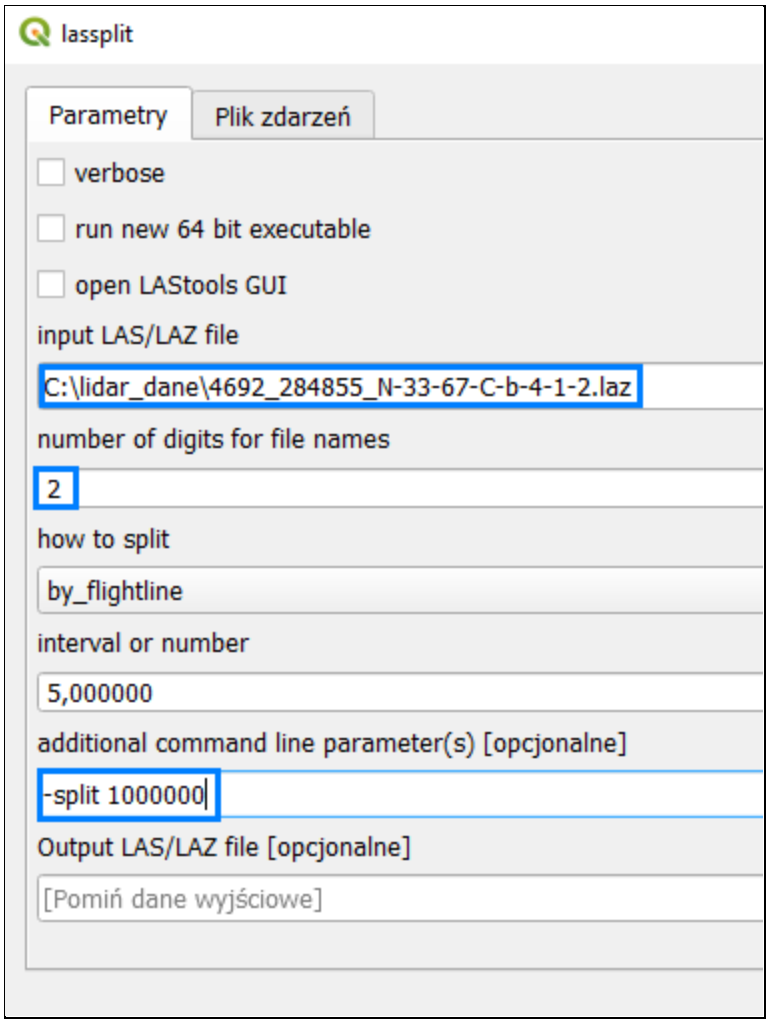
Prezentowany wycinek zestawienia zawiera kody liczbowe przypisane poszczególnym elementom skanowanego obszaru.

W ramach ćwiczenia spróbujemy wyselekcjonować z chmury punkty reprezentujące rzędną terenu. Według klasyfikacji ukrywają się on pod liczbą “2”. Informacja ta będzie nam potrzebna na kolejnym etapie przetwarzania.

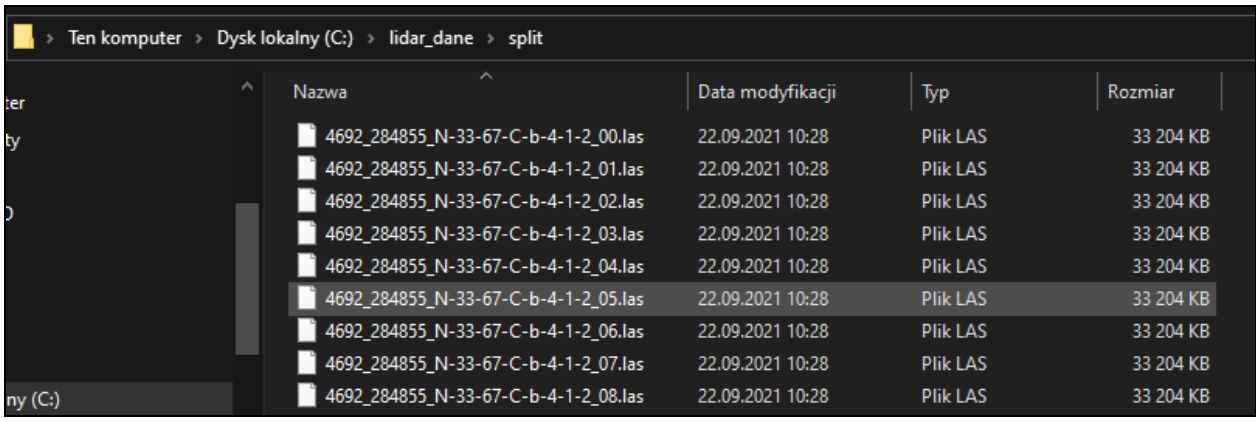
Ze względu na ograniczenia wersji bezpłatnej musimy podzielić nasz zbiór na mniejsze części o liczebności nie przekraczającej jednego miliona. Wykorzystamy do tego funkcję *lassplit*, którą mogą Państwo wywołać z panelu *Algorytmów Processingu*:



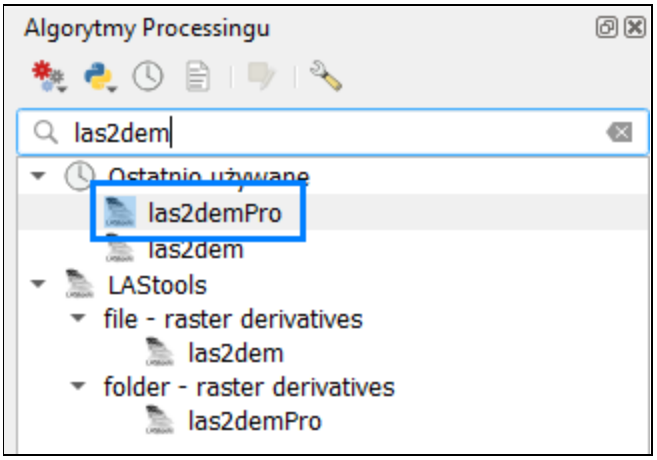
Uruchamiamy algorytm. W oknie dialogowym wskazujemy ścieżkę do pliku, zmniejszamy parametr *number of digits* do 2 (jest to liczba cyfr przyłączanych do nazwy rozszczepionych plików wynikowych), wskazujemy linię przelotu jako podstawę rozdzielenia i dodajemy dodatkowy parametr *-split 1000000* (dzięki temu podzielimy obraz źródłowy na części po 1 mln obiektów każda). Klikamy na *Uruchom*.



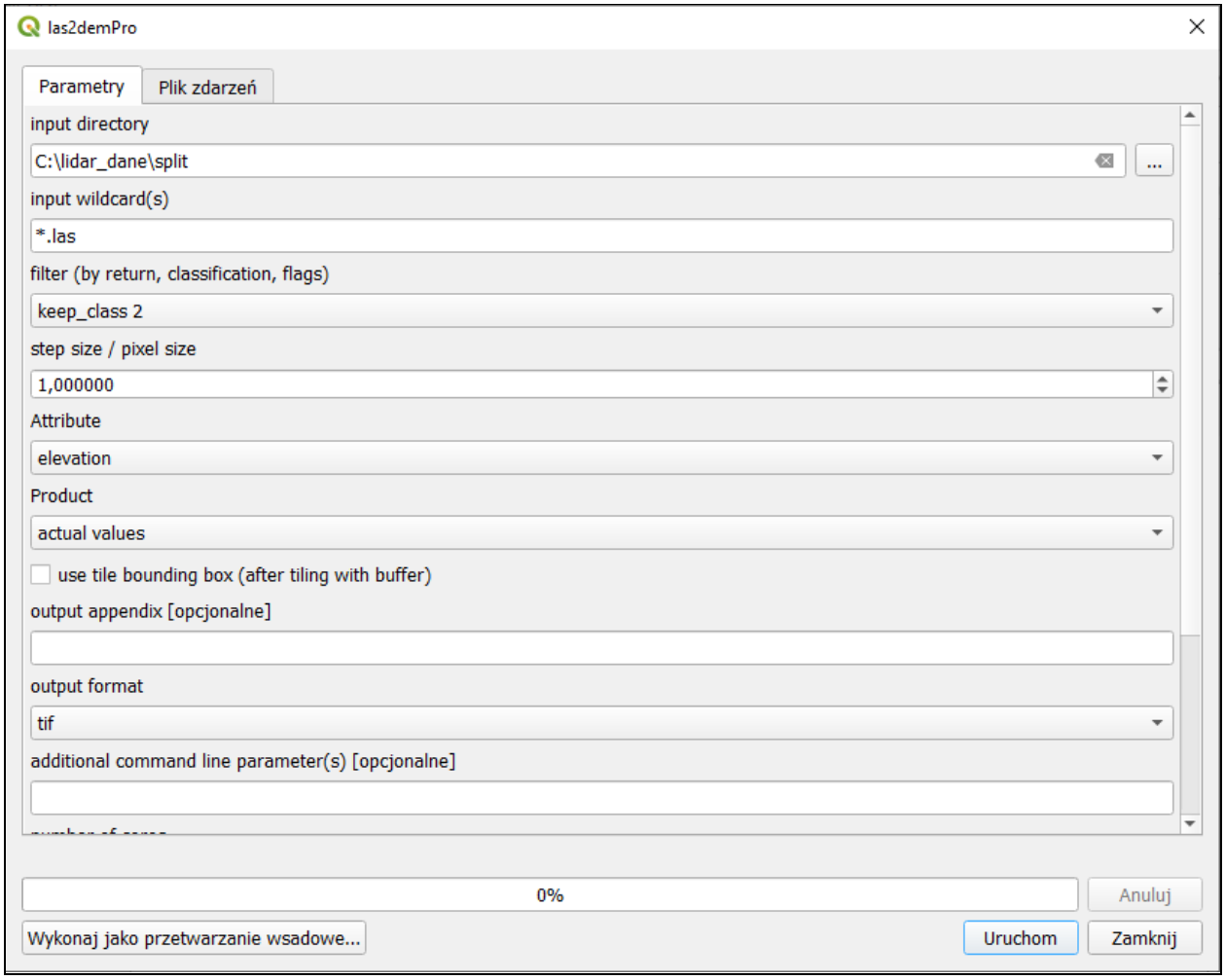
W wyniku działania w folderze *C:\lidar_dane* pojawi się kilkanaście nowych plików z rozszerzeniem .las. Należy utworzyć nowy subkatalog *C:\lidar_dane\split* i przenieść do niego nowe pliki.



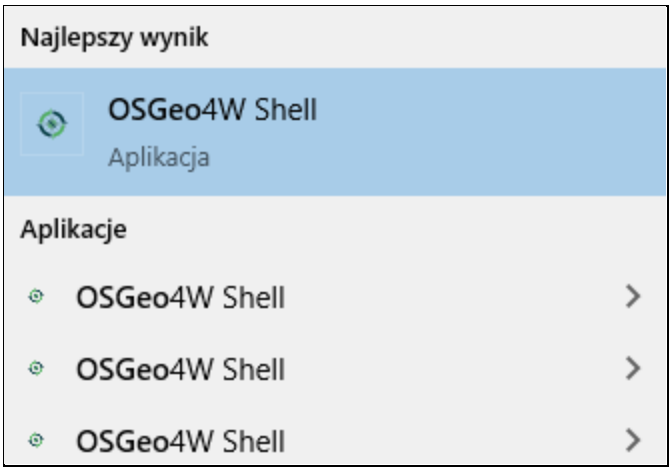
Następnie spróbujemy przetworzyć pliki na format tif, by w dalszej kolejności połączyć je w jeden obraz wynikowy. Pierwszy etap, tj, konwersję, wykonamy za pomocą narzędzia *las2demPro*:



W oknie dialogowym wskazujemy ścieżkę do subkatalogu *split*, wybieramy format .las, w polu *filter* ustawiamy opcję *keep_class 2*. Dodatkowo jako folder docelowy wskażmy ścieżkę *C:\lidar_dane\split*. Pozostałe parametry pozostawiamy bez zmian. Uruchamiamy proces przetwarzania.



W efekcie w subkatalogu *split* pojawią się nowe pliki w formacie .tif. Do ich złączenia możemy użyć narzędzi biblioteki *GDAL* uruchamianych z poziomu powłoki OSGeo4W. Aby uruchomić powłokę, otwieramy menu *Start* systemu *Windows* i wpisujemy “OSGeo”. Wybieramy aplikację z listy rekordów spełniających kryteria zapytania:



Proces łączenia rozbijemy na dwa etapy, aby uniknąć ewentualnych błędów w przetwarzaniu. Najpierw utworzymy raster wirtualny, będący mozaiką plików .tif, a następnie przekonwertujemy go do formatu GeoTIFF. Komenda pierwsza prezentuje się następująco:

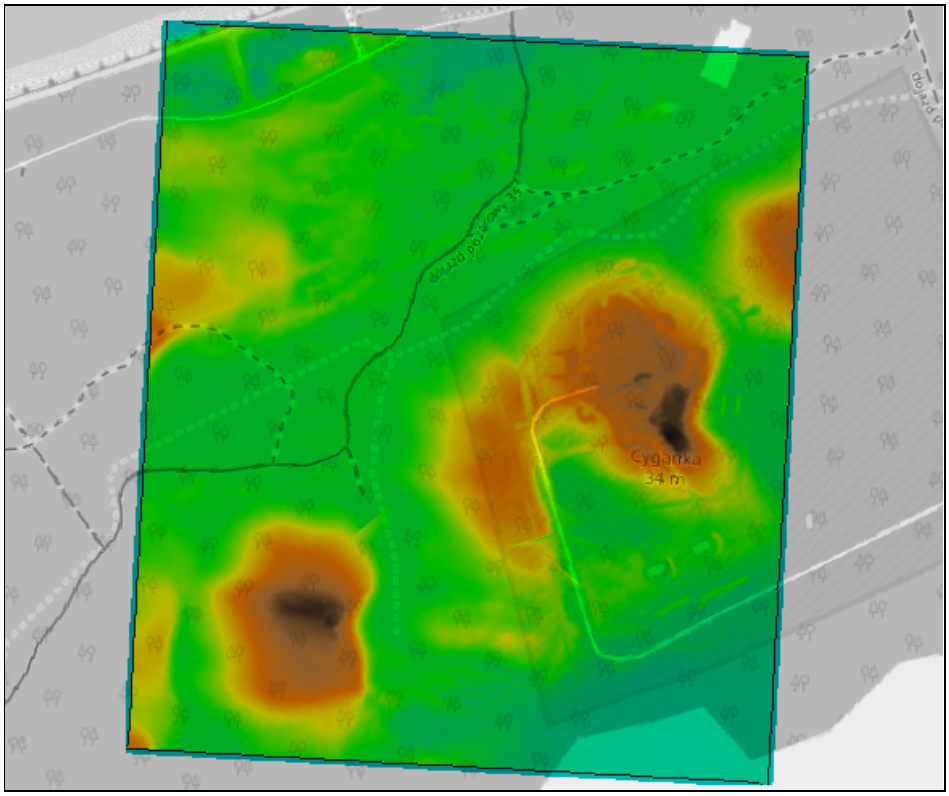
gdalbuildvrt C:\lidar_dane\merged.vrt -a_srs EPSG:2180 -r cubic -tr 0.5 0.5 "C:\lidar_dane\split*.tif"

W tym przypadku skorzystaliśmy z narzędzia *gdalbuildvrt* (pełna dokumentacja na stronie https://gdal.org/programs/gdalbuildvrt.html). Do wyrażenia dodaliśmy dodatkowe parametry: -a_srs ustanawia układ współrzędnych pliku wynikowego, -r cubic ustawia metodę resamplingu, zaś -tr określa rozdzielczość piksela w metrach. Plik wynikowy zostanie utworzony w ścieżce *C:\lidar_dane\merged.vrt*.

Kolejna komenda przekonwertuje raster wirtualny na plik w formacie GeoTIFF. Wszystkie niezbędne parametry wprowadziliśmy w wyrażeniu pierwszym, w związku z czym nie musimy ich powtarzać na etapie ostatecznej konwersji. Wystarczy więc krótkie polecenie:

gdal_translate -of GTiff C:\lidar_dane\merged.vrt C:\lidar_dane\result.tif

Plik wynikowy znajduje się w katalogu *C:\lidar_dane* pod nazwą result.tif. Możemy dodać go do programu QGIS, przeciągając bezpośrednio z okna katalogu do okna mapy:



Automatyzacja czynności analitycznych przy użyciu Modelarza Graficznego

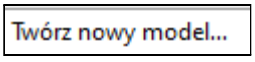
Modelarz graficzny  to narzędzie, które dzięki przejrzystemu interfejsowi graficznemu pozwala w prosty i szybki sposób przygotować własne złożone skrypty, składające się z wymaganej liczby algorytmów uruchamianych sukcesywnie według ustalonej przez użytkownika kolejności. Każdy z etapów procesu przetwarzania danych traktowany jest jako oddzielne ogniwo łańcucha, natomiast relacje między poszczególnymi ogniwami reprezentowane są za pomocą symbolu linii.

Wyróżniamy dwa podstawowe rodzaje “ogniw”: dane wejściowe oraz algorytmy. W skład pierwszej kategorii wchodzą m.in. różne formaty danych przestrzennych oraz tekstowych. Do drugiej natomiast zaliczają się wszelkie algorytmy processingu oraz narzędzia GRASS i SAGA GIS. Dla rozróżnienia dane wejściowe oznaczone są w graficznym widoku modelu za pomocą żółtych prostokątów, natomiast algorytmy - białych:

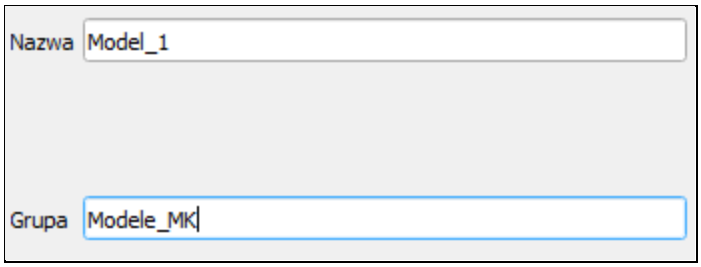
Do korzystania z modelarza nie jest wymagana wiedza programistyczna. Narzędzie pozwala jednak na włączenie do łańcucha przetwarzania własnych skryptów napisanych w języku *Python*. Projekt struktury naszego modelu prezentuje się następująco:

1. Zdefiniowanie warstw wejściowych: raster *Corine Land Cover* oraz granice województw.
2. Wyodrębnienie zaznaczonego województwa.
3. Wykorzystanie wyodrębnionego województwa jako maski do przycięcia rastra.
4. Reklasyfikacja wartości rastra do przy użyciu *Kalkulatora rastra*.
5. Poligonizacja (wektoryzacja) rastrowej warstwy wynikowej.
6. Wyodrębnienie obszarów leśnych ze spoligonizowanej warstwy rastrowej.
7. Użycie warstwy z obszarami leśnymi do obliczenia powierzchni i procentowego udziału określonej kategorii terenu w granicach wybranego województwa.
8. Zastosowanie algorytmu *Zmień Pola* w celu uzyskania "wyczyszczonej" *Tabeli Atrybutów* w warstwie wynikowej.

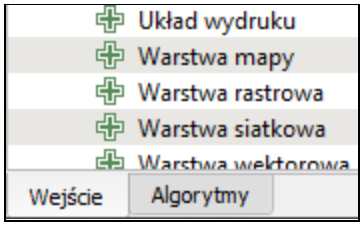
Tworzenie modelu

Modelarz stworzymy w ramach nowego projektu, do którego dodajemy warstwę rastrową CLC_2018.tif z katalogu CLC_2018 oraz dane wektorowe z granicami województw. Zaczynamy od otwarcia okna *Panelu Algorytmów* i kliknięcia na ikonkę  . Z rozwijanej listy wybieramy opcję 

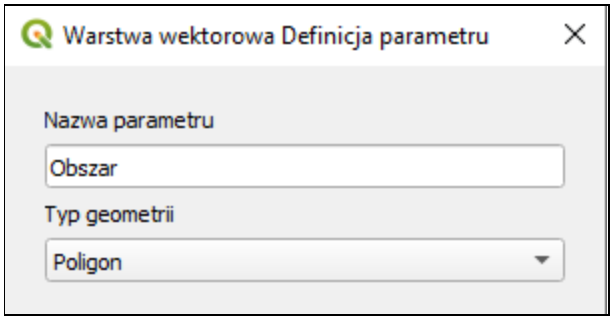
. Przechodzimy do okna kreatora modelarza. Na początku nadajmy naszemu modelowi nazwę oraz stwórzmy dla niego grupę:



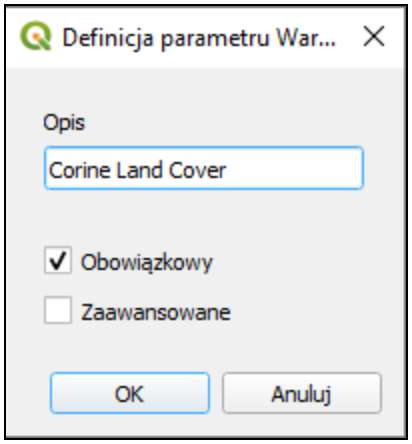
Przejdźmy teraz do okna zlokalizowanego w dolnej, lewej części ekranu. Możemy przełączać widok pomiędzy kategoriami danych wejściowych oraz listą algorytmów:



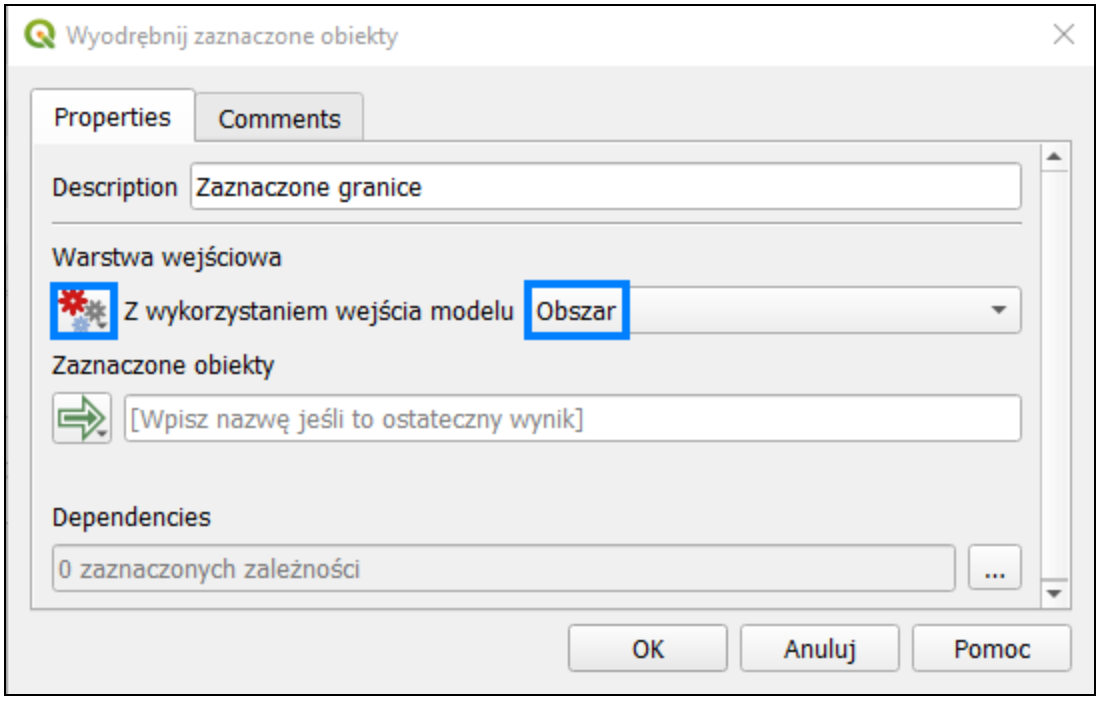
Zatrzymajmy się na chwilę przy kategorii *Wejście*. Do poprawnego działania naszego modelu potrzebujemy dwóch rodzajów danych, tj. warstwy wektorowej oraz rastrowej. Aby dodać warstwę wejścia do modelu, odszukujemy kategorię na liście, klikamy na nią lewym przyciskiem myszy i upuszczamy w oknie po prawej. Po przeciągnięciu otworzy się okno dialogowe, w którym możemy nadać *Nazwę Parametru* (nazwa na określenie tej części procesu - mamy tutaj dowolność) oraz *Typ geometrii* (tutaj warto dopasować opcję do charakteru warstwy - w naszym przypadku będzie to *Poligon*:



Ten element modelu możemy dodatkowo oznaczyć jako obowiązkowy. Klikamy na *OK* i w podobny sposób dodajemy do modelu kolejną warstwę wejściową - tym razem będzie to raster:



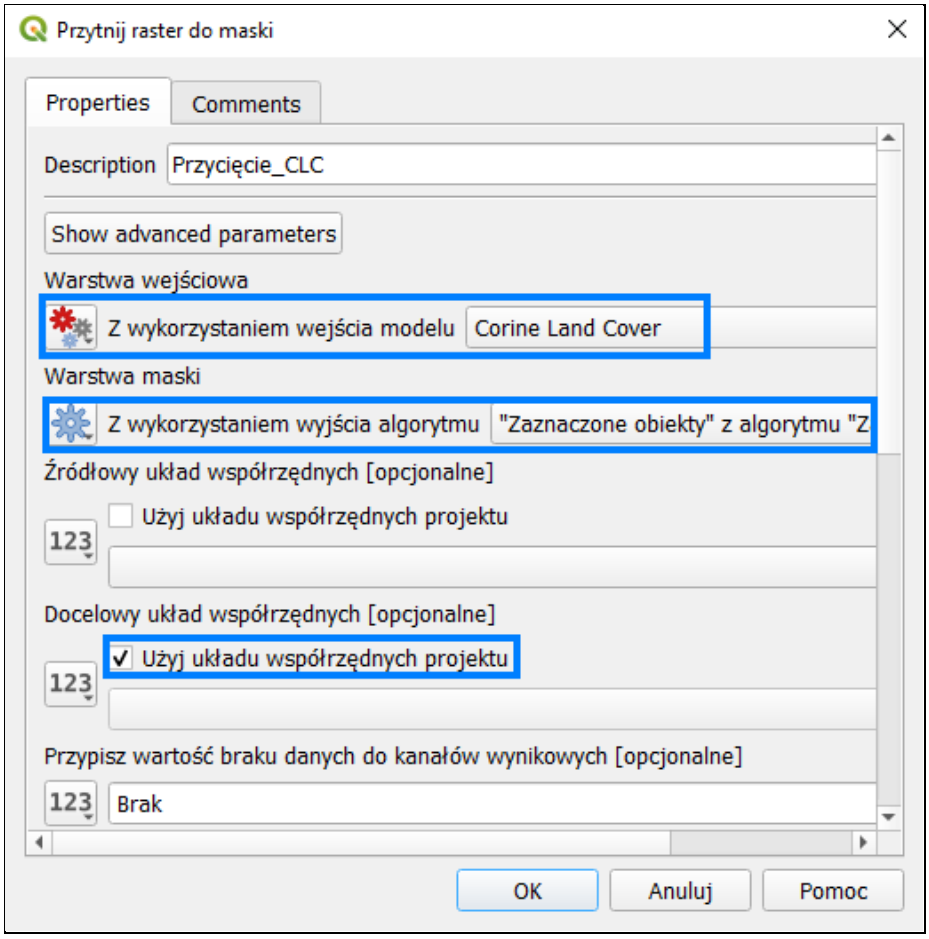
Celem projektowanego modelu jest uzyskanie odczytu procentowej powierzchni wybranej klasy terenu, np. lasów iglastych, dla *wybranego* województwa. Musimy więc zastosować algorytm, który wyodrębni zaznaczony w oknie mapy obiekt i umieści w nowej warstwie tymczasowej. Przechodzimy więc do zakładki Algorytmy i wybieramy z listy Wyodrębnij zaznaczone obiekty. Wypełniamy pola według wzoru:



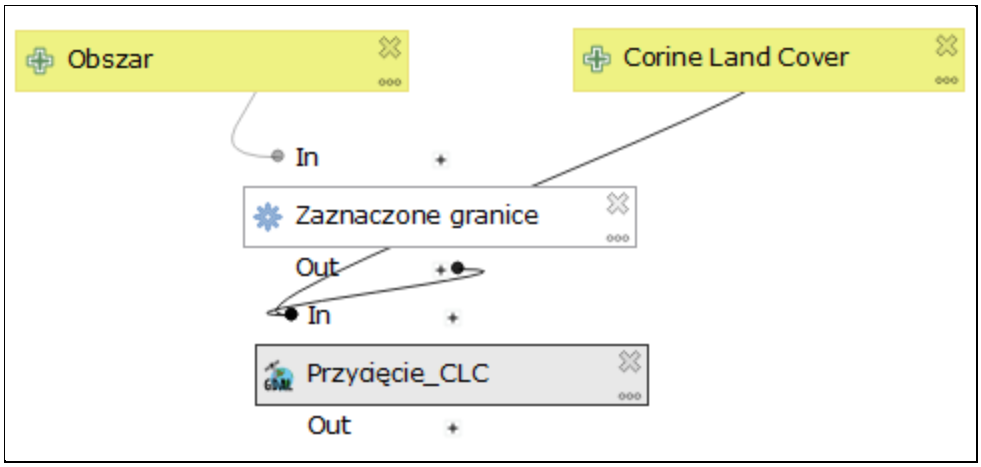
UWAGA! W nowszych wersjach programu QGIS (a do takich zalicza się aktualna wersja LTR) warstwy wejściowe zdefiniowane w modelarzu należy wskazywać po uprzednim wybraniu opcji *Z wykorzystaniem wejścia modelu*. Wyboru dokonujemy poprzez rozwinięcie ikony z kołami zębatymi. Pole *Zaznaczone obiekty* celowo pozostawiamy puste. W innym wypadku warstwa

tymczasowa zostałaby potraktowana przez program jako wynikowa i tym samym dodana do projektu, a tego nie chcemy. Klikamy na *OK* i przechodzimy do edycji kolejnego ogniwa.

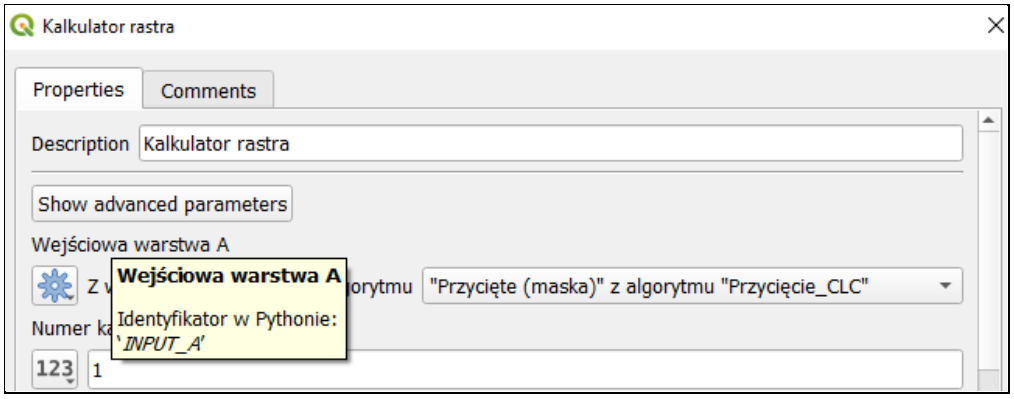
Korzystając z pola *Szukaj* w widoku listy algorytmów, lokalizujemy i przeciągamy do okna po prawej pozycję Przytnij raster do maski. Ustawiamy parametry według poniższego wzorca:



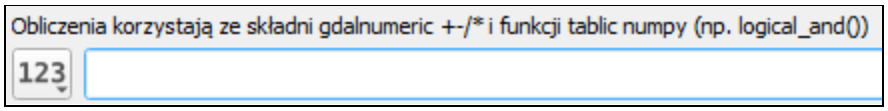
Pozostałe ustawienia pozostawiamy bez zmian. Na ten moment układ naszego modelu powinien wyglądać mniej więcej tak (rozmieszczenie elementów względem siebie można modyfikować, klikając na dany prostokąt lewym przyciskiem myszy i przeciągając go w dogodne miejsce):



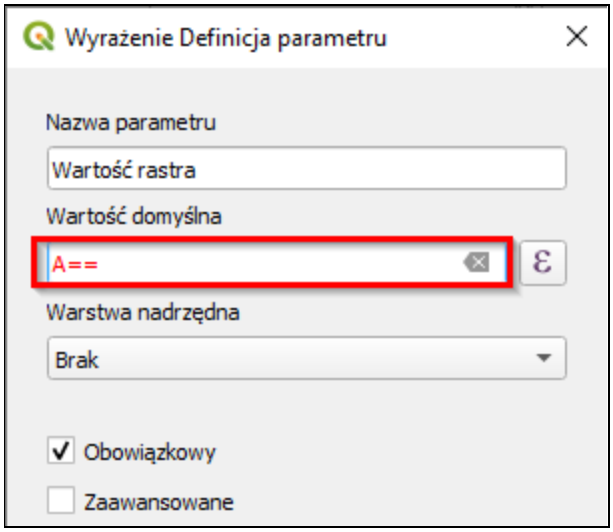
Na rastrze przyciętym do granic województwa należy teraz dokonać reklasyfikacji w taki sposób, by wszystkie piksele zaliczające się do wybranej klasy terenu miały wartość 1, pozostałe zaś - 0. Dodajemy więc do naszego modelu kolejny element, tj. Kalkulator rastra i ustawiamy parametry przekształcenia według następującego klucza:



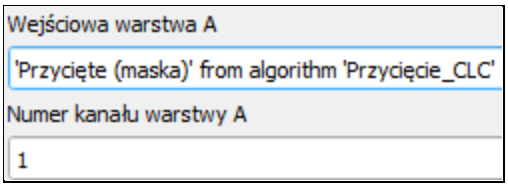
Natomiast pole



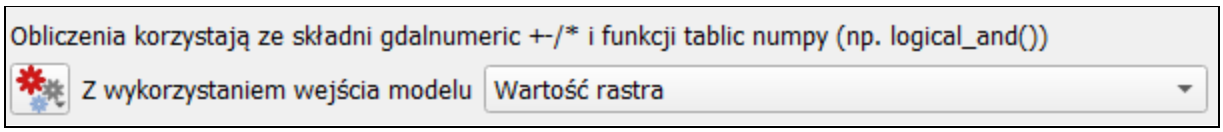
chwilowo pozostawiamy puste. Zamiast wpisywać stałą wartość “podepniemy” pod tę sekcję wyrażenie, dzięki czemu uzyskamy możliwość definiowania klasy rastra jeszcze przed uruchomieniem algorytmu, zwiększając tym samym wszechstronność projektowanego narzędzia. Kliknijmy więc *Anuluj* i z poziomu listy danych wejściowych wybierzmy opcję Wyrażenie. Pola wypełniamy według wzoru:



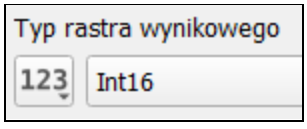
Teraz możemy ponownie dodać do naszego modelu Kalkulator rastra. Pierwsze pola wypełniamy według wzoru:



Natomiast do obliczeń wykorzystamy wcześniej przygotowane wejście modelu z wyrażeniem *Wartość rastra*:

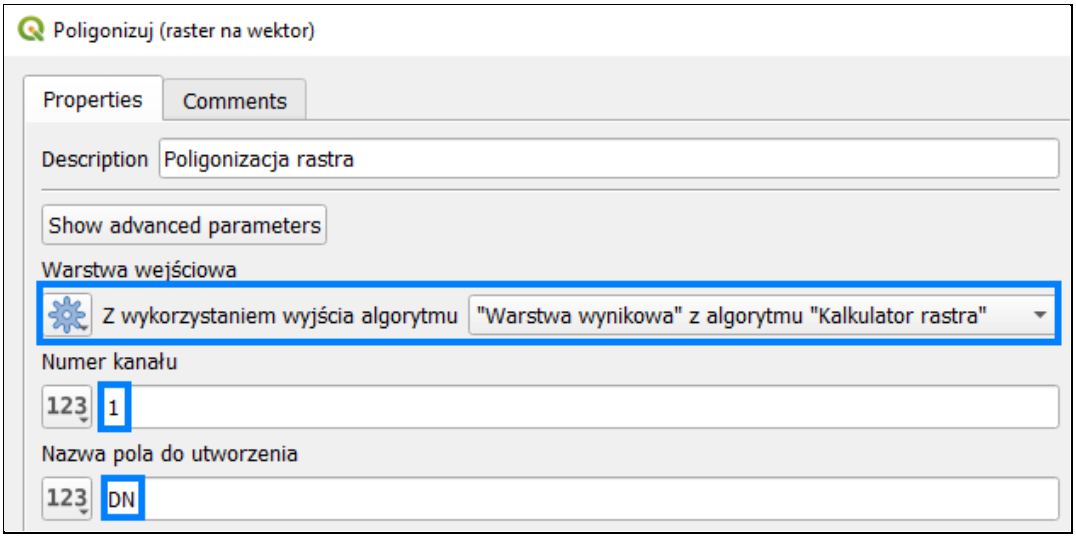


Nasz raster wynikowy zawierać będzie jedynie wartości o charakterze liczb całkowitych, dlatego też zmienimy typ danych wynikowych na *Integer16*:



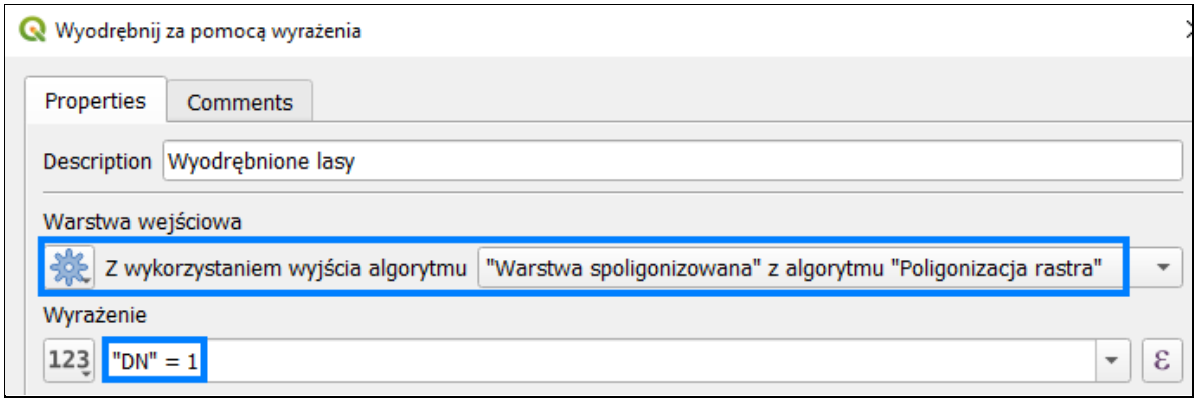
Klikamy na *OK* i przechodzimy dalej.

Wynik reklasyfikacji musimy teraz przetworzyć z postaci rastrowej na wektorową. Wykorzystamy do tego narzędzie Poligonizuj (raster na wektor). Wypełniamy pola:

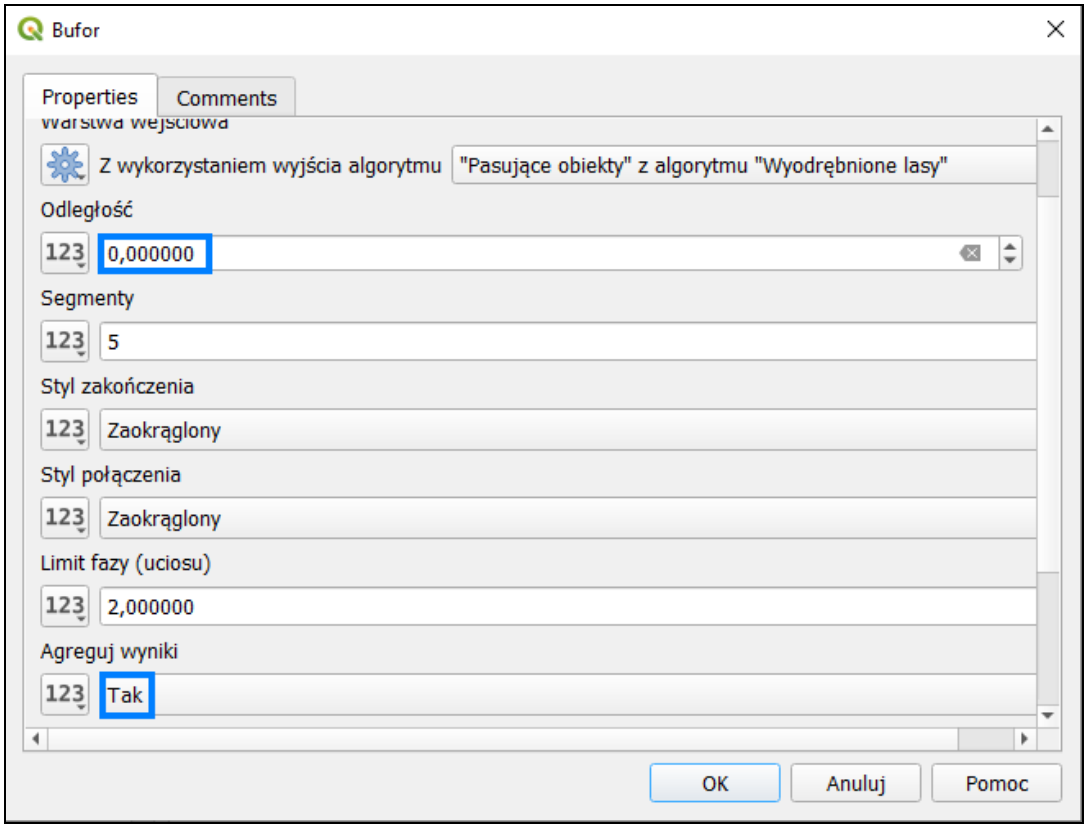


i klikamy na *OK*.

Teraz musimy wyodrębnić jedynie te obiekty, które w warstwie spoligonizowanej przyjmują wartość 1, czyli odpowiadają zasięgowi lasów iglastych. Do projektu modelu dodajemy algorytm Wyodrębnij za pomocą wyrażenia i wypełniamy pola:

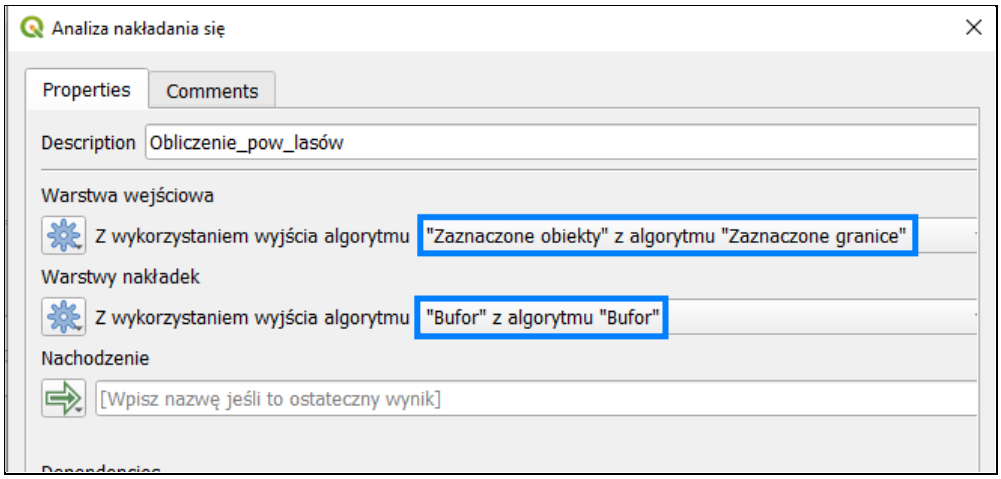


Aby uniknąć ewentualnych błędów w geometrii, zagregujmy dodatkowo obiekty w warstwie *Wyodrębnione lasy*. Wyjątkowo wykorzystamy do tego celu bufor o zerowej odległości z aktywną opcją agregacji:

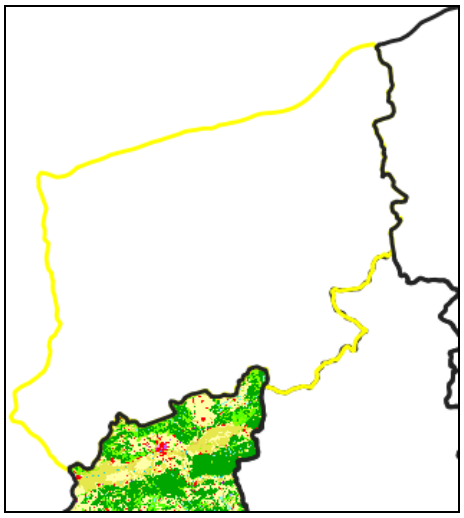


Następnie dodajemy do modelu algorytm 

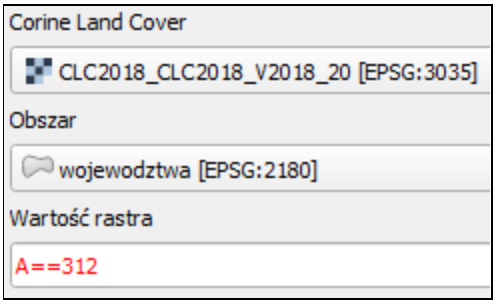
. Wykorzystamy go do obliczenia powierzchni i procentowego udziału wybranej klasy terenu na obszarze określonego województwa. Warstwą źródłową są w tym przypadku *Zaznaczone obiekty* z algorytmu *Zaznaczone granice* (o ile stosowaliście Państwo takie samo nazewnictwo - generalnie jednak chodzi o poligon z wybranym w oknie mapy województwem). Jeśli chodzi o nakładkę, to będzie to *Bufor*::



Tym razem w polu *Nachodzenie* wpiszmy WYNIK. Następnie kliknijmy na , aby zapisać model. Po zapisaniu modelu możemy go uruchomić w celu sprawdzenia, czy wszystkie etapy przetwarzania działają poprawnie. Jeśli tak, w *Panelu warstw* w widoku okna projektu pojawi się nowa warstwa WYNIK. Przed uruchomieniem modelu musimy jeszcze zaznaczyć granice wybranego województwa w oknie mapy. Proponuję *województwo zachodniopomorskie*:



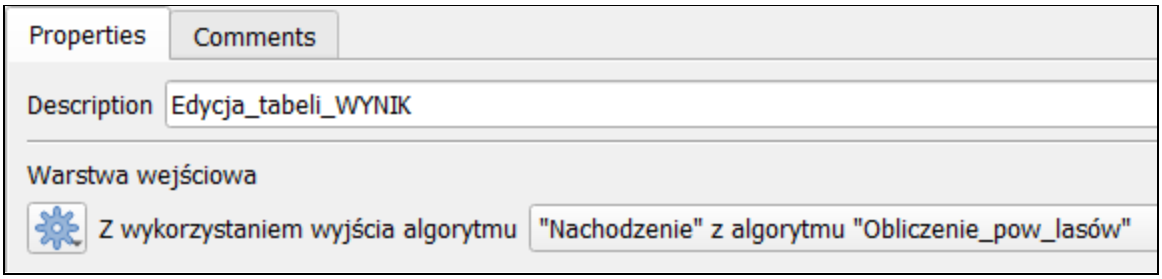
Algorytm uruchamiamy z widoku modelarza klikając na ikonkę zielonego trójkąta na pasku narzędzi, lub wciskając klawisz F5. Po uruchomieniu wyświetli się jeszcze jedno okno dialogowe, w którym musimy wskazać warstwy wejściowe oraz ewentualne parametry przekształcenia (np. wartość rastra do reklasyfikacji - tutaj wybierzmy *312*, by obliczyć powierzchnię lasów iglastych):



Klikamy na *Uruchom* i spokojnie obserwujemy przebieg procesu. Po jego zakończeniu przechodzimy do widoku projektu i sprawdzamy *Tabelę Atrybutów* warstwy WYNIK. Jeśli przetwarzanie przebiegło poprawnie, na końcu listy pól powinny pojawić się atrybuty z obliczoną powierzchnią i wartością procentową:

Bufor_area	Bufor_pc
4899896110,1...	21,400024285...

Zasadniczo na tym moglibyśmy zakończyć przetwarzanie, niemniej *Modelarz graficzny* daje nam możliwość edycji pól znajdujących w poszczególnych warstwach. Warto z niej skorzystać i uporządkować dane w warstwie wynikowej. Wracamy więc do widoku *Modelarza* i dodajemy jeszcze jeden element, mianowicie algorytm Zmień pola. Po jego dodaniu wyświetli się nowe okno dialogowe. Na początek zmieniamy nazwę parametru i wskazujemy warstwę źródłową:



Następnie wczytujemy pola z warstwy WYNIK:

Po wczytaniu przystępujemy do edycji zawartości okna *Mapowanie pól*. Korzystając z ikonki  po prawej stronie okna, usuwamy wszystkie pola poza

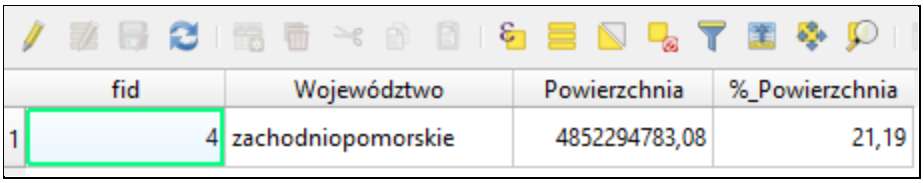
	Source Expression		Name	Typ	Długość
0	123 fid	ε	fid	Liczby całkowite (integer - 64bit)	0
1	abc JPT_NAZWA_	ε	JPT_NAZWA_	Tekst (string)	128
2	1.2 Bufor_area	ε	Bufor_area	Liczby dziesiętne (double)	0
3	1.2 Bufor_pc	ε	Bufor_pc	Liczby dziesiętne (double)	0

Teraz możemy zająć się edycją kolumny *Nazwa pola* i wprowadzić własne nazwy dla atrybutów. W przypadku pól _area i _pc pamiętamy o wskazaniu typu *Double* (liczba dziesiętna) i ustaleniu długości (proponuję 10) i dokładności (liczba miejsc po przecinku - sugeruję 2). W moim przypadku całość po "przeróbkach" prezentuje się następująco:

	Source Expression		Name	Typ	Długość	Dokładność	Relacje
0	123 fid	ε	fid	Liczby całkowite (integer - 64bit)	0	0	
1	abc JPT_NAZWA_	ε	województwo	Tekst (string)	128	0	
2	1.2 Bufor_area	ε	Powierzchnia	Liczby dziesiętne (double)	10	2	
3	1.2 Bufor_pc	ε	Powierzchnia_%	Liczby dziesiętne (double)	10	2	

Po wprowadzeniu zmian nadajemy nazwę warstwie wynikowej, np. *Czysty_Output*. Klikamy na *OK* i wracamy do widoku modelarza. Przed zapisaniem modelu wprowadzimy jeszcze jedną modyfikację, polegającą na usunięciu wcześniej uwzględnionej warstwy wynikowej:

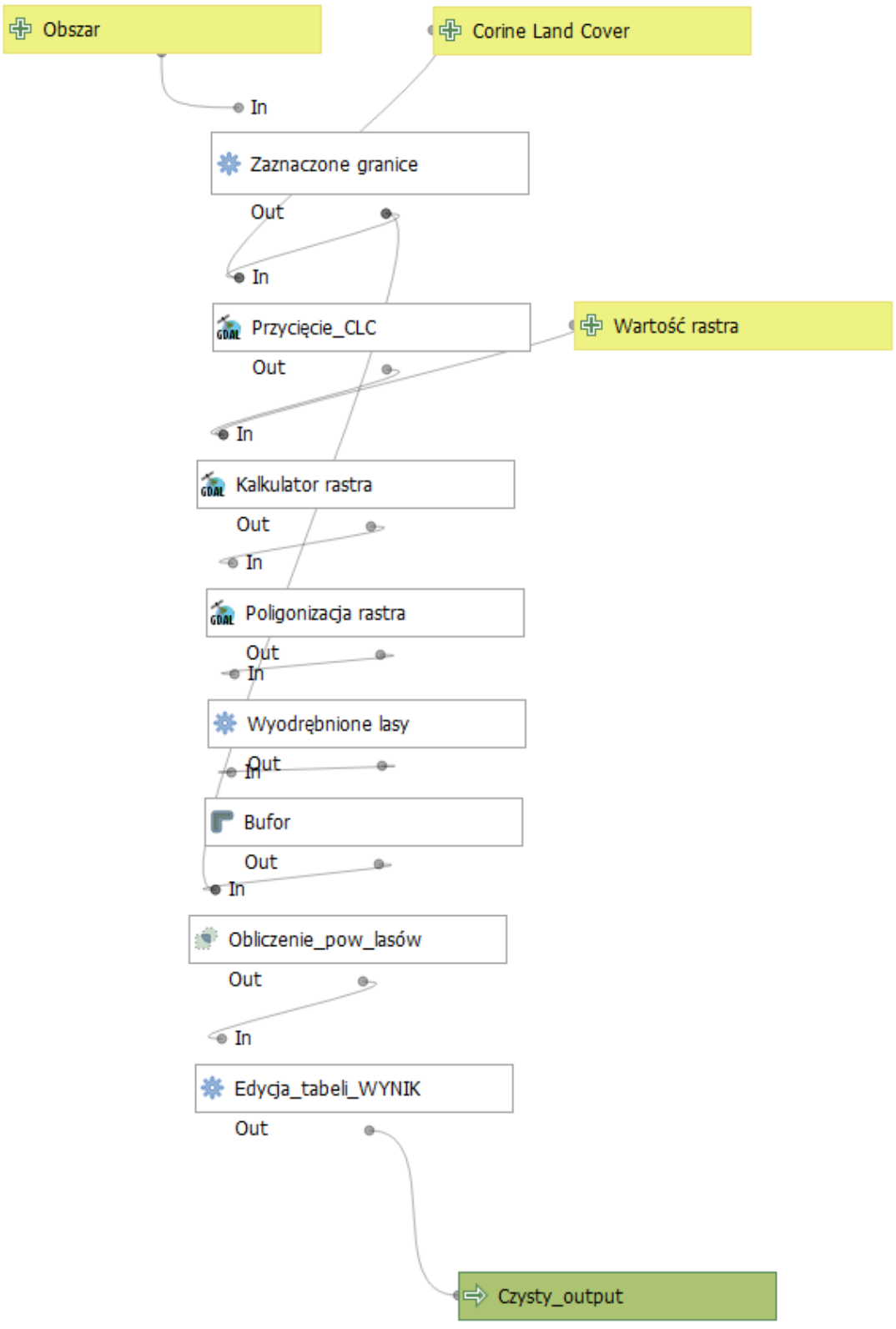
Zapisujemy i uruchamiamy model, a następnie przechodzimy do sprawdzenia *Tabeli Atrybutów* warstwy *Czysty_output*:



	fid	Województwo	Powierzchnia	%_Powierzchnia
1	4	zachodniopomorskie	4852294783,08	21,19

Jak widać na załączonym obrazku, wszystkie zmiany w strukturze tabeli zostały wprowadzone zgodnie z założeniami.

Natomiast całe drzewko algorytmów prezentuje się następująco:


Obszar
Corine Land Cover
In
Zaznaczone granice
Out
In
Przycięcie_CLC
Wartość rastra
Out
In
Kalkulator rastra
Out
In
Poligonizacja rastra
Out
In
Wyodrębnione lasy
Out
In
Bufor
Out
In
Obliczenie_pow_lasów
Out
In
Edycja_tabeli_WYNIK
Out
Czysty_output