Univerzita Palackého v Olomouci

Přírodovědecká fakulta

Katedra geoinformatiky

AUTOMATIZACE ZOBRAZENÍ TURISTICKÝCH TRAS NA MAPÁCH

Diplomová práce

Bc. Jakub POSPÍŠIL

Vedoucí práce Mgr. Radek Barvíř, Ph.D.

Olomouc 2024

Geoinformatika a kartografie

ANOTACE

Diplomová práce se zabývá automatizací vstupních dat reprezentujících značené turistické trasy z databáze **OpenStreetMap (OSM)** do provedení, kdy jsou značené trasy v mapách znázorňovány paralelně podél linií vrstvy komunikací. Typickými příklady map, kde nalezneme více linií znázorněných podélně s liniemi vrstvy komunikací, jsou turistické, cyklistické či průvodcovské mapy.

Teoretická část práce se zabývá rešerší zahrnující vyhledání a hodnocení současně používaných variant znázornění turistických či jiných značených tras v mapách. Dále se rešeršní teoretická část práce zabývá programovacím jazykem **Python** a jeho knihovnami, které umožňují rozšíření základní funkcionality, což umožní tvorbu skriptu v praktické části této diplomové práci.

V praktické části je provedena algoritmizace postupu metody znázornění značených tras podél linií komunikací. Je rovněž zvážena možnost, kdy je liniová reprezentace tras z databáze OSM zobrazena s vrstvou komunikací z jiného zdroje dat, které nemají totožný průběh, například z důvodu generalizace dat pro různá měřítka. Postupně je v praktické části popsána tvorba **Python** scriptu pro program **ArcGIS Pro** od firmy **Esri**. V tomto programu je skript také testován a funkcionalita navrženého postupu je poté demonstrována formou ukázkové turistické mapy oblasti Hrubého Jeseníku.

Pro tvorbu skriptu byl použit program **Visual Studio Code** od společnosti **Microsoft** a testování probíhalo v programech **ArcGIS Pro** a **QGIS**. Výsledná ukázková mapa byla graficky zpracována do finální podoby a připravena k tisku v programu **Adobe Illustrator** z balíčku **Adobe Creative Cloud**.

KLÍČOVÁ SLOVA

Turistika; turistické trasy; automatizace, značení tras, znakový klíč

Počet stran práce: 71

Počet příloh: 6 (z toho 2 volné a 4 elektronické)

ANOTATION

The master thesis deals with the automation of input data representing marked hiking routes from the OpenStreetMap (OSM) database into a form, where the marked routes in the maps are mapped parallel to the road layer lines. Typical examples of maps where more lines are represented parallel to the road layer lines are hiking, cycling or guide maps.

The theoretical part of the thesis deals with research including searching and evaluation of currently used variants of representation of hiking or other marked routes in maps. Furthermore, the research theoretical part of the thesis deals with the Python programming language and its libraries, which allow the extension of the basic functionality, which will allow the creation of scripts in the practical part of this thesis.

In the practical part, the algorithmizing of the procedure of the method of representation of marked routes along the lines of roads is carried out. The possibility is also considered where the line representation of routes from the OSM database is displayed with a layer of roads from another data source that does not have the identical course, for example due to the creation at a different scale. The practical part describes the development of a Python script for ESRI's ArcGIS Pro. The script is also tested in this program and the functionality of the proposed procedure is then demonstrated in the form of a sample tourist map of the Hrubý Jeseník region.

The **Visual Studio Code** program from **Microsoft** was used to create the script and testing was done in **ArcGIS Pro** and **QGIS**. The resulting sample map was graphically processed into its final form and prepared for printing in **Adobe Illustrator** from **Adobe Creative Cloud** package.

KEYWORDS

Hiking; hiking routes; automation, route marking, symbology

Number of pages: 71

Number of appendixes: 6

Prohlašuji, že

- diplomovou práci včetně příloh, jsem vypracoval(a) samostatně a uvedl(a) jsem všechny použité podklady a literaturu.

- jsem si vědom(a), že na moji diplomovou práci se plně vztahuje zákon č.121/2000 Sb. - autorský zákon, zejména § 35 – využití díla v rámci občanských a náboženských obřadů, v rámci školních představení a využití díla školního a § 60 – školní dílo,

- beru na vědomí, že Univerzita Palackého v Olomouci (dále UP Olomouc) má právo nevýdělečně, ke své vnitřní potřebě, diplomovou práci užívat (§ 35 odst. 3),

- souhlasím, že údaje o mé diplomové práci budou zveřejněny ve Studijním informačním systému UP,

- v případě zájmu UP Olomouc uzavřu licenční smlouvu s oprávněním užít výsledky a výstupy mé diplomové práce v rozsahu § 12 odst. 4 autorského zákona,

- použít výsledky a výstupy mé diplomové práce nebo poskytnout licenci k jejímu využití mohu jen se souhlasem UP Olomouc, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly UP Olomouc na vytvoření díla vynaloženy (až do jejich skutečné výše).

V Olomouci dne 9. května 2024 Jakub POSPÍŠIL

Chtěl bych upřímně poděkovat vedoucímu práce Mgr. Radku Barvíři, Ph.D. za pomoc, ochotu, připomínky a konzultace během tvorby této magisterské práce. Velké díky mu patří hlavně za motivaci, časovou flexibilitu a věcné rady při tvorbě této práce. I přes všechny komplikace se podařilo dospět do zdárného konce s pěkným výsledkem. A právě díky myšlence zlepšení turistických map bylo možné vytvořit a zpracovat tuto diplomovou práci a částečně automatizovat proces tvorby turistických map. Velké poděkování rovněž patří mým spolužákům a kamarádům, se kterými jsem svou práci konzultoval a dostával díky tomu zpětnou vazbu a mnoho zajímavých nápadů. Mé poslední díky bych chtěl upřímně vyjádřit mé rodině a přítelkyni za obrovskou podporu na cestě celým mým studiem.

UNIVERZITA PALACKÉHO V OLOMOUCI

Přírodovědecká fakulta

Akademický rok: 2022/2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Jakub POSPÍŠIL**
Osobní číslo:	**R220011**
Studijní program:	**N0532A330009 Geoinformatika a kartografie**
Téma práce:	**Automatizace zobrazení turistických tras na mapách**
Zadávající katedra:	**Katedra geoinformatiky**

Zásady pro vypracování

Cílem práce je automatizovat postup zpracování vstupních dat reprezentujících značené trasy z databáze OpenStreetMap (OSM) do typického provedení, kdy značené trasy jsou znázorněné paralelně podél linie samotné komunikace. V rešeršní části student vyhledá a ohodnotí současně používané varianty znázornění turistických či jiných značených tras v mapách. Následně provede algoritmizaci postupu metody znázornění značených tras podél linií komunikací pro vybraný GIS software. Student zváží i variantu, kdy je liniová reprezentace tras z OSM databáze zobrazena s vrstvou komunikací z jiného zdroje dat, které nemají totožný průběh. Funkcionalita navrženého postupu bude demonstrována tvorbou ukázkové mapy.
Text práce student zpracuje v souladu se závaznou šablonou pro kvalifikační práce KGI. O diplomové práci student vytvoří webovou stránku a poster. Celou práci (text, přílohy, výstupy, zdrojová a vytvořená data, poster a web) odevzdá student v digitální podobě na datové úložiště katedry. Do evidence STAG student odevzdá úplný text práce s přílohami, které určí vedoucí práce. Fyzicky student odevzdá výtisk posteru ve formátu A2 a přílohy určené vedoucím práce.

Rozsah pracovní zprávy:	**max. 50 stran**
Rozsah grafických prací:	**dle potřeby**
Forma zpracování diplomové práce:	**elektronická**

Seznam doporučené literatury:

[1] TEULADE-DENANTES, J., MAUDET, A. & DUCHENE, C. (2015). *Routes visualization: Automated placement of multiple route symbols along a physical network infrastructure.* Journal of spatial information science, 2015(11), 53–79. DOI: 10.5311/JOSIS.2015.11.230.
[2] CHILTON, S. (2017). Neocartography and OpenStreetMap. In *The Routledge handbook of mapping and cartography*, 276–284. Routledge.
[3] KSHETRI, T. B. (2021). PySLD: An Open-source Python Package for Generating the Symbology of Geospatial Data. In *FOSS4G – ASIA 2021.*
[4] HOCHMAIR, H. H., ZIELSTRA, D. & NEIS, P. (2013). Assessing the completeness of bicycle trails and designated lane features in OpenStreetMap for the United States and Europe. In *Transportation Research Board Annual Meeting.*
[5] VOŽENÍLEK, V. (2002). *Diplomové práce z geoinformatiky.* Olomouc: Univerzita Palackého v Olomouci.

Vedoucí diplomové práce: **Mgr. Radek Barvíř, Ph.D.**
Katedra geoinformatiky

Datum zadání diplomové práce: **9. prosince 2022**
Termín odevzdání diplomové práce: **9. května 2024**

L.S.

doc. RNDr. Martin Kubala, Ph.D.
děkan

prof. RNDr. Vilém Pechanec, Ph.D.
vedoucí katedry

V Olomouci dne 1. září 2023

OBSAH

POUŽITÁ LITERATURA A INFORMAČNÍ ZDROJE
PŘÍLOHY

SEZNAM POUŽITÝCH ZKRATEK

Zkratka	Význam
ČR	Česká republika
ČÚZK	Český úřad zeměměřický a katastrální
DMR5G	Digitální model reliéfu 5. generace
GIS	Geoinformační systémy
KČT	Klub českých turistů
OSM	OpenStreetMap
RGB	Red Green Blue
SHP	Shapefile
TK ČÚZK	Terminologická komise ČÚZK
UNWTO	Světová organizace cestovního ruchu
WJS	The Wall Street Journal

ÚVOD

Značené trasy provázejí lidstvo již od dob kočovnictví a počátku obchodu. I v této době bylo nutné se umět orientovat jak v reálném světě, tak i v mapě, což právě pojem „značené trasy“ spojuje. Značení v tomto slova smyslu je spíše míněno jako označování cest a měst pomocí cedulí a ukazatelů s šipkami, kudy se lidé musí vydat. Jednou z historicky nejvýznamnějších obchodních cest je jednoznačně 9 tisíc kilometrů dlouhá, tzv. „Hedvábná stezka/cesta“, vedoucí z Číny až ke Středozemnímu moři. Obecně je známá jako první globální obchodní cesta v historii lidstva, která měla rovněž daleko větší význam, než jen obchod se zbožím (UNWTO, 2024). Nejen na této trase bylo potřeba se orientovat, ale s postupně přicházejícím větším množství map bylo třeba řešit také značení již zmíněných tras a cest na mapách. V dnešní době je značení takovýchto tras širším tématem, jelikož se problematika značení rozvinula hlavně o volnočasové aktivity, typicky o turismus a cykloturismus. Česko má jednu z nejhustších sítí turistických tras na světě (TÉMA, 2020). Nachází se zde přes 40 000 kilometrů značených tras, které jsou vzájemně dobře propojeny a značeny (Klub českých turistů, 2024). Faktem je, že český systém značení byl inspirací i pro další země. Inspirovalo se například Rumunsko, Polsko nebo Bulharsko (Česká televize, 2016). Český systém značení má ale mnohem větší než jen evropský přesah. Trasy s českými značkami se v dnešní době objevují už i v Brazílii (iDNES.cz, 2016) nebo Mongolsku (Klub českých turistů, 2023). Český systém značení tras a mapy s tímto značením tedy bezesporu patří mezi jedny z nejlepších systémů na světě.

Hlavní myšlenkou této práce je pak optimalizace procesu tvorby map se značenými trasami podél linií vrstvy komunikací, jejichž dobrým příkladem jsou mapy od Klubu českých turistů (KČT). Při procesu tvorby mapy je kladen důraz na co největší automatizaci a ušetření času při samotné tvorbě.

Tato diplomová práce je navázána na zjištěnou problematiku v bakalářské práci autora, kdy výstupem byl tištěný turistický průvodce po méně známých místech Olomouckého a Moravskoslezského kraje. Při tvorbě této publikace byl zjištěn problém nefungujících offsetů tras a jejich vykreslování v prostředí ArcGIS Pro a tato problematika zákresu značených tras musela být poté řešena časově náročným a neoptimálním způsobem.

Tato práce je tedy i osobní motivací autora přijít s optimálním řešením v desktopovém prostředí pro budoucí tvorbu map se značenými trasami. Další motivací je zlepšení dovedností v oblasti programování a programovacího jazyka Python, který je nedílnou součástí ArcGIS Pro a je rovněž vybraným programovacím jazykem při procesu optimalizace zákresu tras při tvorbě map.

1 CÍLE PRÁCE

Hlavním cílem této diplomové práce je automatizace postupu zpracování vstupních dat, které reprezentují značené trasy, z databáze OpenStreetMap (OSM) do typického provedení, kdy jsou značené trasy znázorňovány do podoby linií vedoucích podél linie komunikace například v turistických mapách. Během tvorby této práce je kladen důraz na co největší optimalizaci a ušetření času pro uživatele GIS, kteří se s touto problematikou často setkávají.

Rozpracování cílů

Pro teoretickou a praktickou část diplomové práce byly vymezeny níže zmíněné cíle. Teoretická část se zabývá zjištěním stavu řešené problematiky, konkrétněji pak rešerší variant značení jak turistických, tak i dalších tras v mapách. Dále jsou v této části zkoumány možnosti vykreslení tras a potenciální způsob řešení problematiky. Rešerše se také zabývá možnostmi a knihovnami programovacího jazyka Python a podpůrných knihoven, díky kterým bude provedena praktická část této práce.

Praktická část se zabývá algoritmizací postupu metody znázornění značených tras podél linií komunikací v desktopovém prostředí GIS. Postupně je zde popisován proces algoritmizace a optimalizace provedení. V praktické části je také řešena možnost vykreslování liniových tras, kdy jsou data z databáze OSM zobrazena s vrstvou komunikací z jiného zdroje dat, které nemají totožný průběh. Příkladem jiného zdroje dat je topografická databáze Data50.

Výsledkem této práce je Python skript pro ArcGIS Pro, který umožní uživatelům optimálně nastavit offsety pro vstupní vrstvu značených tras při tvorbě map v desktopovém prostředí. Finální funkcionalita scriptu je demonstrována tvorbou ukázkové turistické mapy oblasti Hrubého Jeseníku.

Součástí zásad pro vypracování této práce je také povinná příloha v podobě posteru ve formátu A2 a tvorba webové stránky v souladu s pravidly dostupnými na stránkách Katedry geoinformatiky Univerzity Palackého v Olomouci.

2 SOUČASNÝ STAV ŘEŠENÉ PROBLEMATIKY

Prvním a hlavním tématem rešerše je samotný pojem turismus, jeho definice, možnosti, limity a podoby. V druhé části jsou popsány současné podoby značených tras v mapách. Další část rešerše je věnována programovacímu jazyku Python a jeho knihovnám, které umožňují rozšíření funkcionality například o prostorové funkce, a také současným možnostem vizualizace. Část této rešerše je postavena na základu položeném v bakalářské práci autora (Pospíšil, 2022), v jejíž rešerši je řešena problematika turistických průvodců související právě se značením tras v mapách.

Jedním z hlavních pojmů této práce je *mapa*, kterou je nutno si na začátek definovat. Slovník terminologické komise Českého úřadu zeměměřického a katastrálního (TK ČÚZK) uvádí definici slova mapa jako *„zmenšený generalizovaný konvenční obraz Země, kosmu, kosmických těles nebo jejich částí převedený do roviny pomocí matematicky definovaných vztahů (kartografických zobrazení), ukazující prostřednictvím metod kartografického znázorňování polohu, stav a vztahy přírodních, sociálně-ekonomických a technických objektů a jevů“* (TK ČÚZK, 2020).

Spojení map a značených tras je v dnešní době hodně asociováno právě s turistikou a volnočasovými aktivitami. Klíčovým úkolem turistických map je pomoc uživateli dostat se z jednoho bodu do druhého. Pro správnou orientaci využívají turisté v mnoha místech mimo jejich obvyklé prostředí právě mapu, nebo turistického průvodce, a to jak v tištěné, tak mobilní/online formě.

Reilly a kol. (2005) uvádějí, že se zdá být nepravděpodobné, že by papírové mapy mohly být překonány mapami na malých obrazovkách z hlediska statického kontextu a vizuální prezentace, a to i přes nedávné slibné snahy o zlepšení zobrazovaných map na mobilních obrazovkách. Od roku 2005 se mapy v elektronických zařízeních vyvinuly tak, že jsou lidmi používané na denní bázi. To ale neznamená, že by klesl význam tištěných map, které jsou dodnes hojně využívané. The Wall Street Journal (2023) tvrdí, že jsme svědky určitého trendu, kdy tištěné mapy nabývají zpět svou popularitu. Své tvrzení dokládají na počtu prodaných map od roku 2020. Ačkoliv rozdílů, výhod a nevýhod mezi tištěnými a online mapami je mnoho, tato diplomová práce se zabývá problematikou, jejíž výsledek lze využít při tvorbě a práci s oběma typy těchto map.

2.1 Turismus

2.1.1 Úvod do turismu

Světová organizace cestovního ruchu (UNWTO) vykládá pojem turismus neboli cestovní ruch jako *„sociální, kulturní a ekonomický fenomén, který zahrnuje pohyb lidí do zemí nebo míst mimo jejich obvyklé prostředí za osobními, obchodními i profesními účely. Lidé provozující turismus se nazývají návštěvníci (také turisté nebo výletníci; rezidenti nebo nerezidenti) a cestovní ruch souvisí s jejich aktivitami, z nichž některé zahrnují výdaje na cestovní ruch“* (UNWTO, 2008). Touto definicí se také tato diplomová práce primárně řídí. Slovník Univerzity v Cambridge, jeden ze světově nejpoužívanějších uvádí poněkud kratší a méně vysvětlující definici turismu. V jeho definici je uvedeno, že se jedná o poskytování služeb turistům, jako je doprava, ubytování nebo zábava (Cambridge University Press & Assessment, 2024). Tyto dvě definice se navzájem doplňují. První definice poukazuje na fakt, že lidé nedílnou součástí turismu jsou také výdaje návštěvníků. Definice z Cambridge Dictionary naopak poukazuje na fakt, že turismus by bez poskytování široké škály služeb návštěvníkům nemohl fungovat. Ačkoli se počátky turismu datují již do prehistorické doby,

turismus, jak ho známe v dnešní době se začal vyvíjet na počátku 20. století (StudyCorgi, 2022) a tento trend pokračuje dodnes. Hunt a Layne (1991) ve své publikaci uvádí, že právě touha po poznání, nových zkušenostech, aktivitách a stylu života byla právě základním stavebním kamenem pro pojem „new tourism“ (Hunt and Layne, 1991), který je znám taktéž pod označením "masový turismus". Rostoucí rozvoj turismu se mimo jiné dá přisuzovat rapidnímu rozvoji dopravní infrastruktury ve světě a s tím souvisejícím snižujícím se nákladům na dopravu. Typickým příkladem rozvoje dopravy společně se snižujícími se náklady jsou v dnešní době nízkonákladové lety. Předchozí příklad rozvoje dopravy potvrzují ve své publikaci Sezgin s Yolalem v roce 2012: „*V prvních letech dvacátého století se cestovní ruch stále rozšiřoval v důsledku rostoucího bohatství, zájmu, odchodů a zlepšení dopravy. Pokrok v dopravě umožnil lidem cestovat ve velkém*“ (Sezgin a Yolal, 2012). Nízkonákladové společnosti a jejich lety se také, minimálně v Evropě, podílejí na rozvoj turismu a většího přísunu turistů do oblíbených lokalit. Tento výrok potvrzuje i článek zkoumající vliv nízkonákladových leteckých společností na cestovní ruch ve Španělsku (Rey a kol., 2010).

2.1.2 Značení tras

Značení tras, cest a stezek hraje pro lidstvo nezastupitelnou roli, která sahá hluboko do historie. Bezpochyby nejznámější historickou trasou byla Hedvábná stezka. Jednalo se o síť obchodních tras, které procházely napříč celou Asií ze Středomoří do Číny a Indie, představovala také informační a kulturní most mezi Evropou a Dálným východem (Synek, 2022). Hedvábná stezka tedy nesloužila jen k přepravování zboží a vzácných komodit, ale díky neustálému pohybu a mísení obyvatelstva docházelo k rozsáhlému přenosu znalostí, myšlenek, kultur a víry, což mělo hluboký dopad na dějiny a civilizace euroasijských národů (Zhaowen, 2023).

Se založením Moravskoslezského sudetského horského spolku v roce 1881 je datován i počátek horské turistiky v oblasti Jeseníků. S rozvojem turismu a horské turistiky obzvláště pak ve 20. století přišlo také značení turistických tras. Trasy začaly být značeny i pro účely volnočasové rekreace, což se později projevilo i v mapové tvorbě. První turistická trasa byla v Čechách vyznačena již v květnu 1889. Vedla ze Štěchovic u Prahy ke Svatojánským proudům (Kadlčáková, 2016). V roce 2024 to bude již 135 let od první značené turistické trasy. Úplně nejstarším turistickým značením v českých zemích je ale tyčové značení na hřebenech Krkonoš. Toto značení je dokonce znázorněno i na mapách Krkonoš ze 17. století (Drahný, 2006).

Za pouhou jednu dekádu od první vyznačené trasy v Čechách došlo nejen k rozšíření oblasti značených tras na území Hrubého Jeseníku, Rychlebských hor a Kralického Sněžníku, ale také ke zhruba pětinásobku vytvořených turistických chodníků v této oblasti. V tu dobu se jednalo přibližně o 480–500 kilometrů vyznačených cest (Glonek, 2019).

V tomto období bylo německými spolky vybudováno přibližně 4 000 km značených tras (Havelka, 2008). Nicméně dodnes nejdůležitějším milníkem, který přetrvává v historii turistiky a značených tras v českých zemích je založení Klubu českých turistů (KČT) 11. června 1888 v čele s předsedou Vilémem Kurzem st., Františkem Čížkem a Vratislavem Pasovským. KČT si vybral způsob značení pomocí pásových značek místo složitějšího tvarového značení využívaného například německými spolky v pohraničí (Šafránek, 2010). Do roku 1915 se však využívaly ke značení jen červené značky. Od této zásady bylo později upuštěno a KČT zavedl vícebarevné značení, jak ho známe dnes. Jedná se o již známou čtveřici barev (Klub českých turistů, 1916). Tomáš Brabec ve své bakalářské práci zjistil,

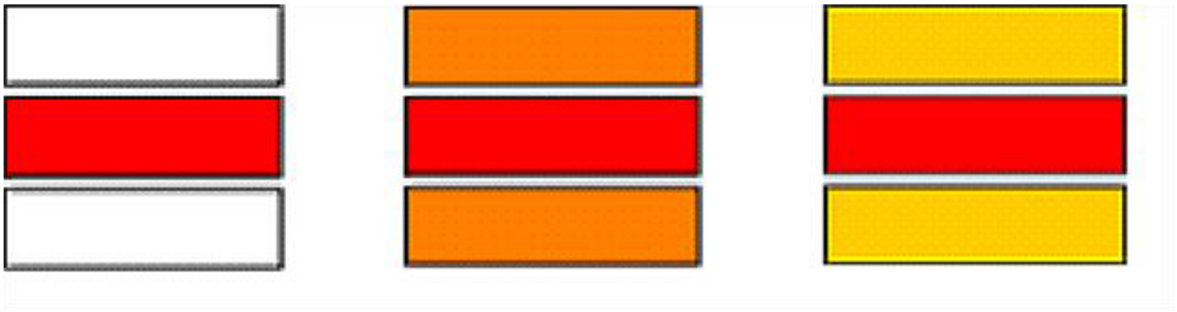
že hlavním důvodem byla již hustá síť červeně značených tras, které se často křižovaly, což vyvolávalo ve výletnících zmatek (Brabec, 2014).

Během období První světové války nastal útlum turismu, který ale po roce 1918 opět nabral na obrátkách právě v době, kdy dochází k obrovskému rozvoji turismu, k čemuž také přispělo například rozšíření automobilismu a dopravní infrastruktury (Historie KČT). Právě díky tomuto rozvoji bylo v roce 1920 již kolem 25 000 km značených tras především v Československu (Brož, 1938). Obecně platí trend, v období mezi světovými válkami probíhá více turistického mapování a rozšiřuje se síť tras. Díky Mnichovské dohodě nicméně KČT, v té době KČST, přišel o více než 15 000 km tras (Historie KČT). Po Druhé světové válce byly organizovány značkařské brigády s cílem zrušit tvarové značení vytvořené původními německými spolky na našem území (Šafránek, 2010). V období mezi lety 1947 a 1951 bylo zorganizováno přibližně 140 takových akcí s průměrnou účastí 20 osob a dohromady bylo vyznačkováno přes 9 000 km tras (Hubička, 1963). V dalších letech rozšířil KČT svou působnost také do oblasti lyžařského a cyklistického značení. Byly vydány metodické publikace, jako je například *Technika značení turistických cest* (Hubička, 1955). Zároveň vznikly nové druhy značek (místní, naučná, koncová, odbočka a další), které jsou používány dones.

Mnohem více do hloubky se těmto tématům věnuje bakalářská práce Petry Mokrošové z Vysoké školy ekonomické v Praze na téma *Vývoj turistických tras a značení na území České republiky* a bakalářská práce Tomáše Brabce z Univerzity Karlovy na téma *Klub českých turistů v letech 1888–1918*.

Nutno také podotknout, že bez určitých pravidel by nemohlo značení vznikat. V případě České republiky stojí za systémem turistického značení Klub českých turistů, který rozlišuje 3 základní druhy turistického značení (obr. 1):

- pěší značení,
- lyžařské značení,
- cykloznačení.



Obrázek 1 – Systém turistického značení v ČR

Značení jednotlivých tras má svá pravidla, na kterou odkazuje příručka vytvořená KČT (Klub českých turistů, 2013). Ve čtyřicetistránkové publikaci jsou podrobně popsána pravidla, jenž musí být dodržována při značení či udržování turistických tras. Například pěší trasy jsou na území České republiky značeny pásovými značkami, které se skládají ze tří vodorovných pruhů. Prostřední pás, který určuje barvu značené trasy, je červený, modrý, zelený nebo žlutý. Oba krajní pruhy jsou bílé a mají za úkol značku více zviditelnit (Inovace předmětu Pěší turistika, 2023). Tvar a orientace jsou v tomto případě rovněž důležité. Samotná pásová značka je tvořena čtvercem o rozměrech 10 × 10 cm. V ostrých lomech cesty nebo při jejím odbočení na jinou komunikaci je pásová značka doplněna na šipku. Zároveň se také na společných úsecích několika značených tras používá vícebarevná značka oddělovaná bílými pruhy. (Klub českých turistů, 2023). Pod KČT spadá rovněž

správa lyžařských tras a cyklotras. Mezi další značené trasy na našem území patří trasy pod hlavičkou Evropské unie. V případě cyklotras jsou v České republice značeny čtyři Evropské cyklotrasy EuroVelo (EV4, EV7, EV9 a EV13). Evropská unie má stejně tak jako KČT svůj vlastní systém značení (European Commision, 2019). Dalším mezinárodně dobře známým stylem značení je využití tří barev (modré, červené a černé) například k vyjádření náročnosti sjezdovek v zimních střediscích.

Nejdůležitější roli hrají v těchto případech nepochybně barvy, které vyjadřují určitou symboliku tras. V ČR se setkáváme se čtyřmi základními barvami, které jsou vázány na jejich kategorii a význam:

- červená,
- modrá,
- zelená,
- žlutá.

Červená barva je primárně využívána pro dálkové a hřebenové trasy, modrá pro trasy významnějšího charakteru nebo regionální trasy, zelená označuje tzv. místní trasy provázející po blízkém okolí a žlutou barvou jsou značeny spojovací trasy a zkratky. V případě žluté trasy ale neplatí vždy pravidlo, že je ta nejkratší (Tříska, 2021).

Barvy nicméně hrají důležitou roli i v ostatních zemích světa. Zatímco Český systém značení je dělen na čtyři barvy podle kategorie a významu trasy, například v Rakousku jsou trasy značeny jen třemi barvami (modrou, červenou a černou) podle náročnosti trasy. Modrá barva reprezentuje méně náročné trasy, které mohou být místy úzké a strmé. Naopak černá barva reprezentuje trasy lemované strmými svahy, jejichž součástí mohou být ocelová lana nebo jednoduché lezecké pasáže. Ve Finsku jsou tyto tři barvy využívány z hlediska výškového rozdílu na trase a náročnosti orientace. Černá Hora také využívá tyto 3 barvy a dělí trasy na jednoduché, středně náročné a náročné bez další specifikace. V Saské části Německa se využívají stejné 4 barvy, jako v případě Česka. Rozdíl je v klasifikaci. Modrá barva je využívána k označení dálkových tras, červená označuje lokální, spojovací a okružní trasy. Zelená a žlutá barva je pak využívána ke značení naučných stezek. V Litvě a Lotyšsku rozlišují modrou a oranžovou barvu na základě toho, kudy trasa vede a zda je součástí evropské dálkové trasy E9 nebo E11 – podél Baltského moře má značka modrou barvu (E9), a v případě lesních turistických tras se používá barva oranžová (E11). Rumunsko využívá červenou, žlutou a modrou barvu podle významnosti trasy. Tvar značky je zde také důležitý. Trojúhelník značí trasy v údolí, kruh okružní trasy a kříž je využíván k označení spojovacích tras. Barvy na značkách ve Španělsku pak poukazují na vzdálenost trasy. Zelená barva je využívána pro trasy do 10 km, žlutá do 50 km a červená pro trasy nad 50 km (European Ramblers Association, 2022).

Význam barev se tedy v různých zemích od sebe liší. Někde barvy označují náročnost tras, někde zase kategorii, významnost nebo délku. V některých případech barva nehraje vůbec žádnou roli a důležitosti naopak nabývá tvar značky, jak je již zmíněno v odstavci výše. V dalších případech tvar nesehrává žádnou roli a jedná se pouze o způsob zákresu, jako je tomu například v Kanadě, kde jsou používány barevné čtvercové značky s bílou šipkou. Dokonce i v Evropě nalezneme mnoho různých způsobů zákresu značek. Ve Španělsku a Portugalsku se můžeme setkat se značkami, které jsou vykresleny pouze dvěma liniemi připomínajícími polskou vlajku (CK Mundo, 2024), ve Slovinsku se pak můžeme orientovat pomocí červených terčíků a bílým středem (Slovenian Tourist Board, 2024). Další způsob značení najdeme v Norsku, kde jsou trasy často značeny červeným písmenem T nebo jednoduchou modrou linií (European Ramblers Association, 2022).

2.2 Turistické mapy

Turistická mapa je dle TK ČÚZK definována jako „*tematická mapa obsahující objekty, jevy a jejich charakteristiky důležité pro turistiku*“ (TK ČÚZK, 2020). Turistické mapy jsou nedílnou součástí turismu a zároveň i jejich produkce je v oblasti kartografické tvorby silně zakořeněna, což jen potvrzuje tvrzení: „*Turistické mapy jsou jedním z nejběžnějších druhů kartografických produktů*“ (Jancewicz a kol, 2017). Článek Jancewicze a kol. (2017) se zároveň z velké části věnuje řešené problematice v této části rešerše, která je věnována aspektům turistických map, počátkům mapování a dále také typům mapy a znázornění tras na mapách. Vymezení definice je v případě pojmu „turistická mapa“ nejednoznačné. Hlavním důvodem je pojem „turismus“, který lze rozdělit do mnoha kategorií, od kterých se poté odvíjejí účel samotné turistické mapy. Při definici je důležité klást důraz právě na charakteristiku a vlastnosti. Například Kałamucki (2005) ve své publikaci zmiňuje, že jako turistickou mapu lze klasifikovat pouze takové kartografický dílo, jehož obsahem (kromě topografického obsahu) jsou turistické informace o poloze a kvalitativních charakteristikách, vzácněji také kvantitativních, společně s informacemi o infrastruktuře (jak turistické, tak i doplňkové). Kaprowski (1973) dříve ve své publikaci kladl důraz také na fakt, že by turistická mapa měla hlavně umožnit uživateli lepší orientaci v terénu. V tomto případě Kaprowski zdůrazňoval nejen obsah mapy, ale také způsob využití a její funkci. Na základě předchozích definic Jancewicz a Borowiczová (2017) navrhují definici, jejíž součástí je výčet prvků, které by turistická mapa měla obsahovat: „*Turistická mapa by měla mít topografický základ a měla by obsahovat informace o turistických zajímavostech v dané oblasti a rovněž turistickou a dopravní (doplňkovou) infrastrukturu, to vše prezentované pomocí běžných kartografických metod a konvenčních znaků s přihlédnutím na vhodné měřítko a zamýšlené použití mapy.*“ Touto definicí se poté řídí i tato práce.

Turistické mapy se samozřejmě odlišují podle požadavků na jejich obsah. Kritérii mohou být například:

- věková skupina,
- účel mapy,
- druh turistické aktivity,
- měřítko mapy,
- zájmová skupina,
- množství textových polí a informací,
- a design.

Na základě kritérií je poté tvořena samotná turistická mapa. Důsledkem velkého množství kritérií je pak nejednoznačné vymezení samotné definice.

2.2.1 Turistické mapování v Česku

Již před počátky turistického mapování je důležité zmínit obecnou důležitost stezek v historii lidstva. Předchozí část této práce je tomuto tématu věnována v sekci 2.1.2. Počátky turistického značení a mapování sahají již do druhé poloviny 19. století (Ptáček, 2020). Dle Galvasové a kol. (2008) byla právě počáteční fáze cestovního ruchu započata v druhé polovině 19. století a pokračovala až do období První světové války. Lze tedy pozorovat stejný počátek vývoje. V období rozvíjejícího se množství volného času, hospodářského rozvoje a období průmyslové revoluce se cestovní ruch začíná formovat. S vývojem cestovního ruchu obecně rostou potřeby turistů, které je třeba uspokojit.

Vznikají zařízení, infrastruktura a nové profese vázané na cestovní ruch (Galvasová a kol., 2008).

Prvotní rozvoj turistického značení přišel v druhé polovině 19. století, kdy začaly vznikat horské a turistické spolky v pohraničí, které popisuje Klauz (2010). Právě díky těmto spolkům začalo na našem území mapování a značení turistických tras. Díky vzniklým spolkům, a hlavně tedy autoru Johannu Ripperovi, vznikla nejstarší turistická mapa Jeseníků, která vznikla nejspíše ke konci roku 1881 (Glonek, 2009). Rapidní vzrůst značení turistických tras v této oblasti na konci 19. století potvrzuje i druhá mapa vydána tímto spolkem v roce 1892 - Special-Karte der mährisch-schlesischen Sudeten. S dalším mapováním tras a vývojem turismu vznikaly další turistické mapy. V roce 1910 vydal F. D. Martinov v Olomouci mapu Malé Karpaty a Biela hora. Opakovaně vycházela také mapa Neueste Spezialkarte vom Riesengebirge (Nejnovější speciální mapa Krkonoš) v měřítku 1 : 50 000 od nakladatelství Mittelbach v Lipsku. Krkonoše byly po dlouhou dobu středem zájmu jak zahraničních, tak i českých kartografů. J. Ambrož sestavil jedny z nejzdařilejších map Krkonoš. Tyto mapy byly vydávány v měřítku 1 : 50 000 V. Neubertem v období před, i po druhé světové válce. Poslední edice byla vydána v roce 1948. S rozvojem turistických map pomohl také rozvoj map vojenských. Vojenský zeměpisný ústav v Praze vydával od roku 1935 sbírky turistických map. Kvůli politické situaci a topografickému podkladu těchto map byly na počátku padesátých let staženy z prodeje. Jako náhradu vydávalo Státní tělovýchovné nakladatelství pohledové mapy od profesora S. Vorela. Nicméně i v této době byly trasy nadále značeny. Snaha turistů a značkařů vyústila vznikem souboru turistických map, které spadaly pod Ústřední správou geodézie a kartografie. Edice map byly vydávány českými a slovenskými kartografickými podniky až do roku 1990. Po roce 1989 byl obnoven KČT a začal vydávat turistické mapy ve spolupráci s vojenskými organizacemi, jejichž mapy byly 40 let uchovávány v tajnosti (Klub českých turistů, 2012).

V dnešní době jsou v Česku hojně využívané mapy od KČT, jejichž edice pokrývá celé území ČR v měřítku 1 : 50 000. Dalším velkým hráčem na trhu je pak Shocart (v roce 2023 odkoupen slovenskou firmou CBS spol, s.r.o.), který vydává turistické mapy v měřítku 1 : 40 000 a rovněž pokrývají celé území Česka. Shocart zároveň produkuje i mapy vybraných oblastí Slovenska. Dalším vydavatelem tištěných turistických map je Kartografie Praha se svými mapami v měřítku 1 : 50 000. V oblasti mobilních zařízení pak dominují Mapy.cz, které vyvinuly i mobilní aplikaci, pomocí které se dá navigovat i v offline režimu bez nutnosti mobilního signálu. Mezi zahraniční vydavatele turistických map se řadí například Lonely Planet, Marco Polo nebo National Geographic (PublishersGlobal, 2024).

2.2.2 Typy turistických map

Prvotním základem je rozlišit dva pojmy: turistická mapa a mapa turismu. Mapa turismu se zabývá přímo tématikou turismu, například mapa vyjadřující změnu v počtu přijíždějících turistů v různých obdobích roku ve vybraných lokalitách. Tento druh map je využíván primárně odborníky. Naopak turistická mapa slouží právě turistům a umožňuje jim plánovat, číst a orientovat se v prostoru dané lokality. Takovým příkladem může být mapa vybrané oblasti obsahující značené trasy, infrastrukturu (silnice, stezky, železnice, zastávky, nádraží atd.), topografický podklad a další zájmové body, jako jsou například vrcholy hor, památky, kostely, hrady, zříceniny, restaurace a další zajímavosti.

Po rozlišení základních dvou pojmů je nutné se zaměřit na rozlišení samotných turistických map. Důležitým kritériem pro rozlišení typů turistických map je jejich účel, který je úzce spjatý s typem turismu. Nejen typ turismu a účel hrají při klasifikaci velkou

roli. Mezi další kritéria bychom mohli zařadit také měřítko nebo způsob vyjadřovacích metod. Jedním z problémů při klasifikaci turistických map do určitých kategorií je právě turismus, jehož formy se neustále vyvíjí a rovněž vznikají nové, díky kterým je nutné rozdělení do kategorií upravovat. Prvním člověkem, který se pokusil o komplexní klasifikaci turistických map pomocí různých kritérií je profesor K. Trafas (Jancewicz, 2017). Jancewicz a Borowiczová (2017) ve svém článku publikovali tabulku rozdělení turistických map vycházejí právě z předchozí kvalifikace K. Trafase doplněnou o nové formy turismu. Tabulka níže (Tab. 1) zobrazuje dělení turistických map mezi jejichž kritéria se řadí účel mapy, druh turismu a rovněž oblasti, ve kterých k vybraným aktivitám spjatými s turismem dochází. Typů turistických map je mnoho a jejich tvorba je spjatá s potřebami určitých zájmových skupin. Je tedy jasné, že mapa pro cykloturistu má jiná specifika a kritéria při tvorbě než například mapa pro sjíždění řek nebo mapa gurmánské turistiky.

Tab. 1: Klasifikace turistických map podle cíle a druhu cestovního ruchu (Jancewicz, 2017)

<table>
<tr><td rowspan="10">Mapy pro poznávací turismus</td><td colspan="3">obecné turistické mapy</td></tr>
<tr><td rowspan="2">pěší turistika</td><td colspan="2">horská</td></tr>
<tr><td colspan="2">v nížinách</td></tr>
<tr><td rowspan="3">turistika definovaná způsobem dopravy</td><td colspan="2">kolo</td></tr>
<tr><td colspan="2">auto/karavan/motocykl</td></tr>
<tr><td colspan="2">vlak</td></tr>
<tr><td rowspan="15">Mapy pro kvalifikovaný turismus</td><td rowspan="2">lyžařská turistika</td><td colspan="2">běh na lyžích</td></tr>
<tr><td colspan="2">sjezdové lyžování</td></tr>
<tr><td rowspan="4">vodní turistika</td><td colspan="2">kanoistika</td></tr>
<tr><td colspan="2">plachtění</td></tr>
<tr><td colspan="2">potápění</td></tr>
<tr><td colspan="2">rybaření</td></tr>
<tr><td rowspan="3">cykloturistika</td><td colspan="2">silniční</td></tr>
<tr><td colspan="2">horská</td></tr>
<tr><td colspan="2">sjezdová</td></tr>
<tr><td rowspan="3">lezecká turistika</td><td colspan="3">horolezectví</td></tr>
<tr><td rowspan="2">alpské horolezectví</td><td>léto</td></tr>
<tr><td>zima</td></tr>
<tr><td rowspan="2">jezdecká turistika</td><td colspan="2">hory</td></tr>
<tr><td colspan="2">nížiny</td></tr>
<tr><td>speleologická turistika</td><td colspan="2"></td></tr>
<tr><td>orientační mapy</td><td colspan="2"></td></tr>
<tr><td rowspan="8">Mapy pro ostatní druhy turismu</td><td colspan="3">obchodní/podnikatelská turistika</td></tr>
<tr><td colspan="3">sportovní a rekreační turistika</td></tr>
<tr><td rowspan="2">poutní turistika</td><td colspan="2">pěší</td></tr>
<tr><td colspan="2">smíšený druh dopravy</td></tr>
<tr><td colspan="3">geoturistika</td></tr>
<tr><td colspan="3">bioturistika</td></tr>
<tr><td colspan="3">gastronomická turistika</td></tr>
<tr><td colspan="3">(další typy turistiky: alkoturistika, naučná turistika, drogová turistika, urbex…)</td></tr>
<tr><td colspan="4">Městské turistické mapy</td></tr>
<tr><td colspan="4">Mapy připravené pro propagaci a reklamu turismu</td></tr>
</table>

2.2.3 Znázornění tras na mapách

Nejčastějším prostředkem pro vyjádření značených tras na mapách jsou barevně rozlišené liniové znaky. V případě ČR se jedná o čtyři barvy týkající se pěší turistiky společně s dalšími, například cykloturistickými či lyžařskými. I tyto znaky je potřeba v mapě od sebe odsadit tak, aby se při souběhu vzájemně nepřekrývaly. KČT ve svých mapách znázorňuje barvy turistických tras v souladu s reálnými značkami, kdežto v případě cyklotras, které jsou v reálném světě značeny žlutými cedulkami, jsou v mapách KČT značeny fialovou přerušovanou linií. Důvodem je odlišení od liniového znaku žlutých turistických tras. Tento koncept znázornění tras převzaly také Mapy.cz. U znázornění cyklotras se ale parametr

liniového znaku, v některých případech, liší od map KČT. Cyklotrasa s názvem je téměř shodná, tedy přerušovaná linie společně s číslem cyklotrasy. Naopak samotná cyklostezka je pak znázorněná nepřerušovanou linií s číslem.

Ve světě se můžeme setkat s více možnostmi vizualizace nejen volnočasových tras na mapách. Rozdíl oproti mapám od KČT nebo Mapy.cz se projevuje v případě mapového serveru freemap.sk, kde se na první pohled jeví, že barevné určení tras zůstalo podobné. Freemap.sk rozlišuje, stejně jako KČT a Mapy.cz, trasy na regionální a místní pomocí podobného mapového klíče. Regionální trasy jsou vizualizovány liniovým znakem bez přerušení a místní jsou reprezentovány přerušovanými liniemi. U cyklotras jsou například použity za sebou jdoucí tečky. Hlavní rozdíl nastává u barev, kde jsou například barevně rozlišovány i cyklotrasy, což se v případě Mapy.cz nebo map od KČT neděje. Důvodem je, že oproti Česku se cyklotrasy na Slovensku jsou značeny barvami (Slovenský cykloklub Šariš Prešov, 2003).

Značení tras na mapách může být v mnoha případech odlišné. Co je ale spojuje, je vyjádření pomocí liniových znaků oproti bodovému značení v reálném světě.

2.3 Programovací jazyk Python

Python je open source vysokoúrovňový programovací jazyk s dynamickou sémantikou, který je založen na objektově orientovaném programování. Jednoduchá a snadno naučitelná syntaxe klade důraz na čitelnost, čímž také snižuje náklady na údržbu. Python rovněž podporuje rozšíření o knihovny a moduly, což základní funkcionalitu ještě více rozšiřuje (Python.org, 2024). Python je využíván k tvorbě webových a softwarových aplikací, zpracování velkého množství dat, provádění složitých matematických operací, skriptování, strojovému učení a k dalším aplikacím/použitím (W3Schools, 2024). Průzkum Stack Overflow (2018) poukazuje na fakt, že popularita jazyka Python trvale roste od roku 2012.

Alternativ k jazyku Python je více. Mezi největšího soupeře se řadí programovací jazyk Java. Výhodami Pythonu oproti jazyku Java jsou například:

- jednoduchost,
- délka samotného kódu a čas vývoje,
- využití velkými společnostmi,
- syntax a rychlost učení.

Naopak Java je lepší například v:

- rychlosti programu a stabilitě,
- vývoji webových aplikací,
- front-end vývoji,
- a v mobilních zařízeních.

Obecně tedy platí, že výhody Pythonu pramení z jeho jednoduchosti a rychlosti učení. Naopak výhodami jazyka Java jsou hlavně komplexní syntax, rychlost a stabilita. Níže je uveden základní rozdíl syntaxe obou jazyků, kde je viditelná rozdílná délka kódů (IONOS, 2023).

Python:

```
print("Hi! This is Python.")
```

Java:

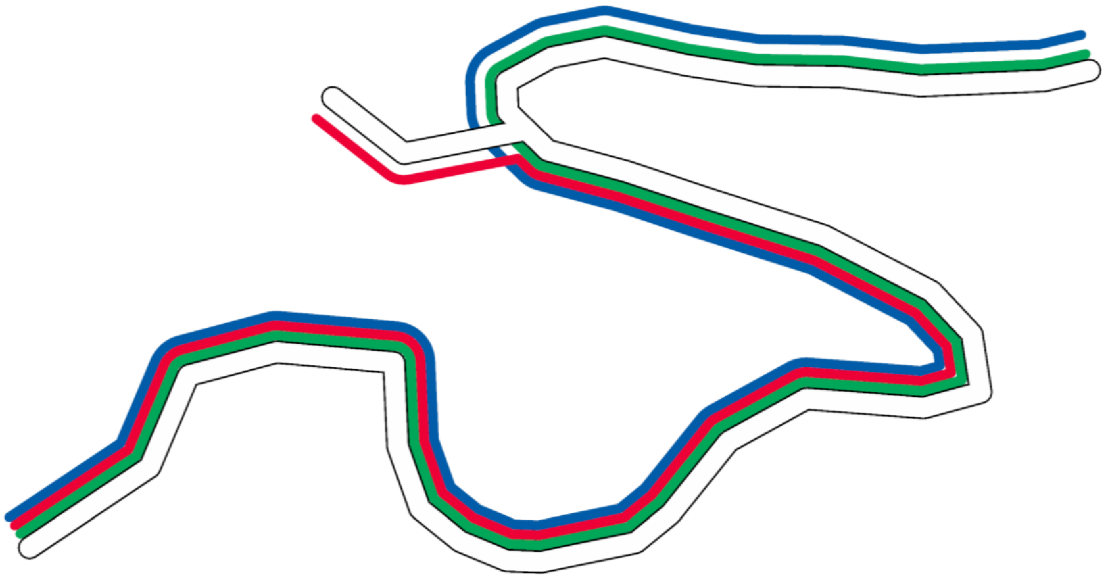
```
class Hello {

  public static void main/String[] args) {

    System.out.println("Hi! This is Java.");

  }

}
```

Mezi další alternativy k Pythonu se řadí JavaScript, který na rozdíl od Javy podporuje podobný styl syntaxe jako python s využitím funkcí a proměnných bez nutnosti definice tříd. V širším srovnání má ale Python výhody v psaní rozsáhlejších programů a dalšímu využití kódu díky objektově orientovanému programování (Python.org, 1997). Mezi další programovací jazyky patří C++, který je sice rychlejší, ale v konečné syntaxi také mnohem delší, než Python (STX Next, 2024). Poslední srovnání proběhne s jazykem C#. Ten je oproti Pythonu typován staticky, což znamená, že má složitější zápis, ale zároveň umožnuje lepší kontrolu chyb a vyšší výkon. C# v dnešní době hodně těží z podpory Microsoftu, kdežto Python je v tomto ohledu silný díky obrovské komunitě vývojářů (Protasiewicz, 2024).

2.4 Optimalizace a současné možnosti vizualizace

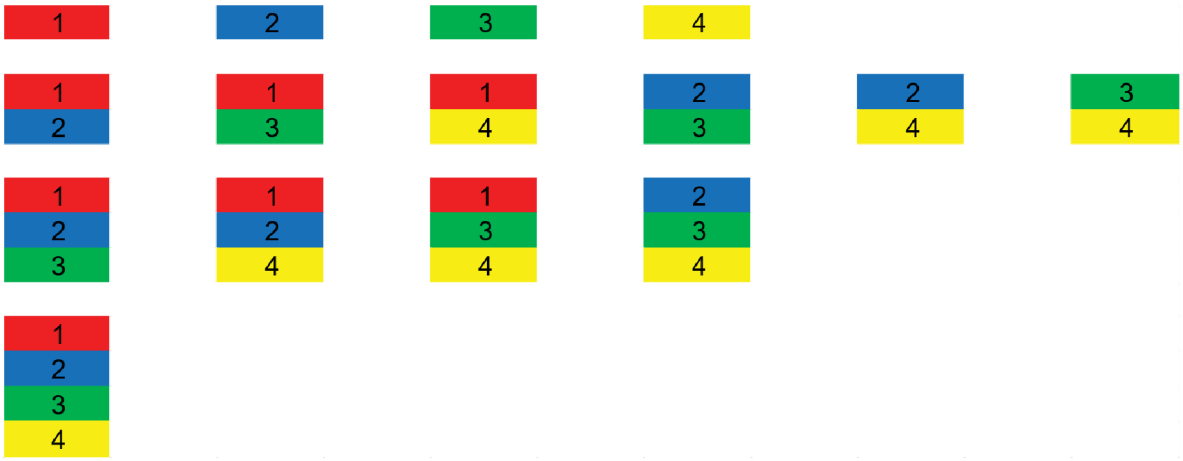
Existuje více možností vizualizace značených tras na mapách. Jedním z přístupů je použití samotného grafického softwaru, typu Adobe Illustrator, CorelDRAW, nebo Inkscape ve kterých je možné vykreslení tras pomocí vektorové grafiky, popřípadě vytvoření nových vrstev jednotlivých tras vedoucích podélně podél linie komunikace. Tato možnost je ale časově velice náročná a neoptimální i z hlediska budoucích úprav dat. Zároveň je zde také možnost, že bude značení velice nepřesné, jelikož neexistuje nástroj, který by dokázal udělat offsety (odsazení linií o předem určené hodnoty) jen na jednu stranu automaticky, a vše se tedy musí dělat ručně pomocí přidávání lomových bodů.

Software ArcGIS Pro nabízí možnost nastavení offsetu jak pro jeden, tak i pro více mapových znaků zároveň. Problém ale nastává u nastavení offsetů u více než jednoho mapového znaku. Tato problematika již byla Pospíšilem (2022) zmiňována dříve. V tomto případě totiž nebyla možnost nastavit mapovým znakům jednotlivé offsety, jelikož se často stávalo, že se mezi znaky linií tvořily mezery. Typickým příkladem je vrstva komunikací společně s třemi podélně vykreslenými vrstvami, jako na obrázku číslo 2, kde je jasně viditelná mezera jak mezi zelenou linií v případě rozcestí, tak i samotné úhybné linie, která není v tomto momentě vykreslena vedle původní komunikace.



Obrázek 2 – Problematika jednotného odsazení

S přidáním většího množství tras tento problém narůstá na objemu. Řešením této problematiky při tvorbě bakalářské práce byla kombinatorika, která byla využita pro výpočet všech variant podélně jdoucích tras, viz. obrázek číslo 3.



Obrázek 3 – Kombinace barev značených tras

Z hlediska kombinatoriky se jedná o jednoduché kombinace, s jejichž pomocí je možné spočítat, kolik možných variant tras může existovat pro definovaný počet barev. V níže uvedené rovnici se objevují dvě proměnné. Písmeno n v našem případě značí samotný počet barev ve vstupní vrstvě a písmeno k pak počet barevných linií vykreslených podél linie komunikací. Díky vzorci číslo 1 jsme schopni vypočítat množství kombinací, ke kterým by při tvorbě mapy mohlo dojít.

$$\binom{n}{k} = \frac{n!}{(n-k)!\,k!} \tag{1}$$

Ze čtyř značených turistických tras (obr. 3) vzešlo celkem 15 barevných kombinací. Po dosazení do vzorce zjistíme, že jednotlivé barvy mohou být 4, kombinací dvou různých barev vzniká 6 možností. Tři jdoucí trasy vedle sebe mají také 4 kombinace. A poslední možnost, kde vedou všechny 4 trasy podél jedné linie, může nastat jen jednou. V případě čtyř barev je tedy možné, že vznikne až 15 barevných kombinací. V případě pěti barev se už jedná o 35 různých kombinací. V případě, že by se barvy podél linií mohly opakovat, jednalo by se o kombinace s opakováním. Pro čtyři barvy by poté vzniklo 69 až kombinací.

V návaznosti na řešení bakalářské práce s patnácti možnými kombinacemi bylo poté základním čtyřem vrstvám přidáno nové pole v atributové tabulce a přiděleny hodnoty podle kombinatorických výsledků. Hodnota 1 v tomto případě znamenala základní offset od původní linie komunikací, jelikož vedla podél linie pouze samotná. Například u hodnoty 134 bylo nutné nastavit znakový klíč tak, aby 3 rovnoběžné linie byly vykresleny červenou, zelenou a žlutou barvou v přesném pořadí a offsetu. Pro kombinaci s červeně značenou trasou bylo tedy zapsáno do atributové tabulky 8 možných kombinací, pro modrou pak už jen 4, pro zelenou dvě a poslední byla neupravená vrstva samotně vedoucí žluté linie. Toto řešení ale není zdaleka optimální, jelikož samotné provedení, editace a nastavení znakového klíče je časově velice náročné. Zároveň s tímto řešením nastává další problém, a to vykreslování tras na obou stranách komunikace v nemálo případech. Ve finální fázi úprav v grafickém softwaru je poté nutné tento problém řešit mazáním jednotlivých liniových znaků.

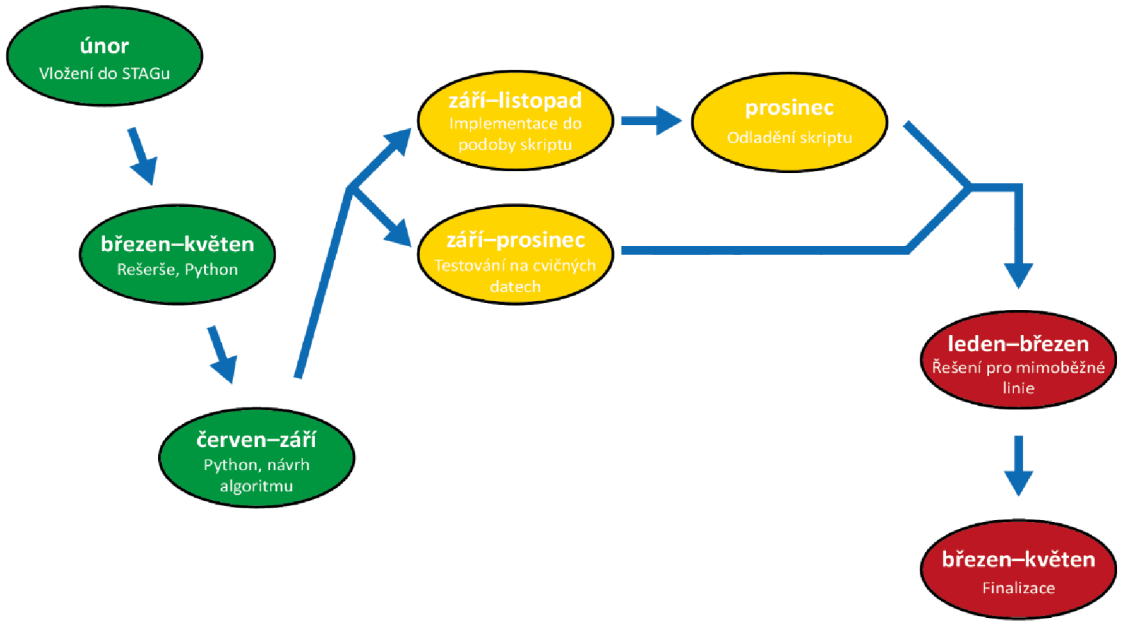
Automatizací procesu umisťování a vizualizace se ve své práci z roku 2015 zabývají Teulade-Denantes, Maudet a Duchêne. V jejich práci je zkoumáno více možností a proměnných, které vstupují do procesu vizualizace značených tras na mapách. (Teulade-Denantes, Maudet a Duchêne 2015). Práce se zabývá nejen řešením souběžně jdoucích linií komunikací a značených tras, ale třeba také řešení v blízkosti křížení linií a jejich umístění závislém na původní vrstvě komunikací. V rámci práce jsou navržena potenciální řešení, která by mohla s realizací této diplomové práce pomoci. Všechny návrhy budou v průběhu realizace praktické části testovány. Maudet, Touya, Duchêne a Picault (2017) ve své práci zabývají algoritmy pro generalizaci vektorových dat v mapách, nicméně určité poznatky z článku mohou rovněž posloužit v řešení praktické části této diplomové práce.

3 METODY A POSTUP ZPRACOVÁNÍ

Před výběrem metod, programů, dat, stanovení harmonogramu a postupu zpracování je stěžejní samotná rešerše dané problematiky. Rešerše je rozdělena na čtyři dílčí témata:

- **turismus,**
- **turistické mapy,**
- **programovací jazyk Python,**
- **Optimalizace a současné možnosti vizualizace.**

Rešerše je tedy zaměřena hlavně na úvod do turismu a značení tras jak v reálném světě, tak v hlavně v mapách samotných. Vizualizace turistických tras na mapách je rovněž důležitým tématem rešerše této práce. Dále je porovnáván programovací jazyk Python s alternativními programovacími jazyky. Python je také vybrán jako nejvhodnější pro tvorbu této práce, jelikož je úzce spjatý s **ArcGIS Pro**. Poslední podkapitola je pak věnována současným možnostem vizualizace, kde je také popsána problematika, kterou se tato práce zabývá. Během zpracování rešerše byl také stanoven časový harmonogram zpracování této diplomové práce (obr. 4). S časovým harmonogramem byl zároveň naplánován rozsah práce a počet jednotlivých výstupů.



Obrázek 4 – Časový harmonogram

Použité metody

Práce byla zpracována jak v praktické, tak i v teoretické rovině. Součástí teoretické části práce je například výše zmíněná rešerše, nebo pak také studium jazyka Python. Další stěžejní částí je studium dokumentací, a to nejen samotného Pythonu, ale hlavně knihovny ArcPy, jejích nástrojů a možností. Tato knihovna je součástí programu ArcGIS Pro a právě s pomocí Pythonu je možné vytvářet skripty a nástroje v běhovém prostředí programu. Další důležitou částí teorie jsou data z OpenStreetMap (OSM). Správné nastudování dokumentace a použití dat z OSM je pro praktickou část této práce stěžejní. Právě díky studování dokumentací byly vymezeny postupné kroky při tvorbě a vybrána data, která byla součástí řešení praktické části.

Praktická část této práce je z většiny tvořena programováním v jazyce Python v prostředí programu **Visual Studio Code** od firmy **Microsoft**. Tvořený skript je průběžně testován v **Arcgis Pro**. V prostředí tohoto programu byla vytvořena sada nástrojů (v ArcGIS Pro pod jménem **toolbox**), které byly testovány a laděny do finální podoby. Postupně byla testována funkcionalita nástrojů z výše zmíněné knihovny a výsledkem jsou čtyři nástroje, které jsou popsány v sekci vlastního řešení této práce, ve dvou toolboxech. Mezi další kroky praktické části patřila tvorba ukázkové mapy. S pomocí vyvinutých nástrojů a správných dat z více datových zdrojů byla vytvořena mapa demonstrující funkcionalitu jednotlivých nástrojů. Design a doladění výsledné mapy byly následně provedeny v programu **Adobe Illustrator**, jež je součástí balíčku **Adobe Creative Cloud**. Součástí praktické části práce je rovněž tvorba **posteru** a **webových stránek**.

Použitá data

Jako hlavní datový zdroj pro tuto práci je **OSM**, tedy databáze volně dostupných geografických dat, která jsou editována a validována samotnými uživateli. Data byla primárně stahována v programu **QGIS** – open source, multiplatformní geografický systém. Do QGISu byla přidána extenze **QuickOSM**, která umožnuje uživateli stahovat data z OSM přímo počítače jako dočasné vrstvy, které mohou být poté uloženy na trvalo. Jelikož jsou součástí této práce primárně data o značených turistických trasách, byl pro stahování dat použit takzvaný **tag** s hodnotou „**route**“, který zajistil **stažení části dat z databáze OSM**. Data s tagem „route“ obsahují **potřebné atributy** (route, colour a osmc_symbo) pro vytvořené nástroje v ArcGIS Pro. Data byla tedy uložena jako shapefile a dále tvořila vstupní vrstvy pro testované nástroje.

Dalším datovým zdrojem je geodatabáze **Data50** od ČÚZK. Tato databáze je poskytována formou otevřených dat od dubna roku 2019 ve formátech SHP (ČÚZK, 2021). Její součástí je řada vrstev vhodných pro tvorbu jak podkladových, tak turistických map. V rámci tvorby byl využit Digitální model reliéfu České republiky páté generace (**DMR 5G**). Pro mapy velkých měřítek je tento model vhodný z důvodu rozměru pixelů, a tedy i rozlišení a přesnosti. V rámci nástroje, který takzvaně **snapuje** linie byla využita i další testovací data a vrstvy.

Použité programy

V rámci zpracování této práce byly využívány zejména dva programy. Prvním z nich je **Visual Studio Code** vyvíjený společností Microsoft. Tento program byl využíván k vytváření a editaci zdrojového kódu vyvíjených skriptů. Práce byla prováděna ve verzi **1.88.1**. Visual Studio Code byl rovněž využit při tvorbě webových stránek o této diplomové práci.

Druhým hlavním programem, ve kterém byly vyvíjené skripty testovány, byl **ArcGIS Pro** od společnosti **Esri** (ve verzi **3.2.2**). V prostředí programu byly tvořeny **toolboxy** a jednotlivé **skripty** byly postupně testovány s přibývající funkcionalitou.

K méně používaným programům při tvoření této práce se řadí **QGIS** ve verzi **3.16**, v jehož prostředí byla, s pomocí pluginu **QuickOSM** (ve verzi **2.1.1**), stahována data z databáze **OSM**.

Finální úpravy výsledné ukázkové mapy spolu s dalšími grafickými prvky byly prováděny ve verzi **2018** programu **Adobe Illustrator**, který se řadí do balíčku programů **Adobe Creative Cloud**. V Adobe Illustrator byla zároveň vyexportována mapa pro následný tisk a vytvořen poster.

Postup zpracování

Hlavní myšlenkou této práce je zjednodušení procesu tvorby turistických map se značenými trasami. Po předchozí bakalářské práci autora byl stanoven cíl a vymezena kritéria. Poté následovalo vymezení definice a zadání práce společně s rozpracováním rešerše dané problematiky. Společně se studováním dokumentací byly započaty práce na praktické části. V úvodu této kapitoly je také vyobrazen časový harmonogram práce (obr.3). Po zpracování praktické části práce bylo přistoupeno k finalizacím, tedy tvorbě webu, posteru a tisku ukázkové mapy.

4 VÝVOJ NÁSTROJE PRO BAREVNĚ ZNAČENÉ TRASY V MAPÁCH

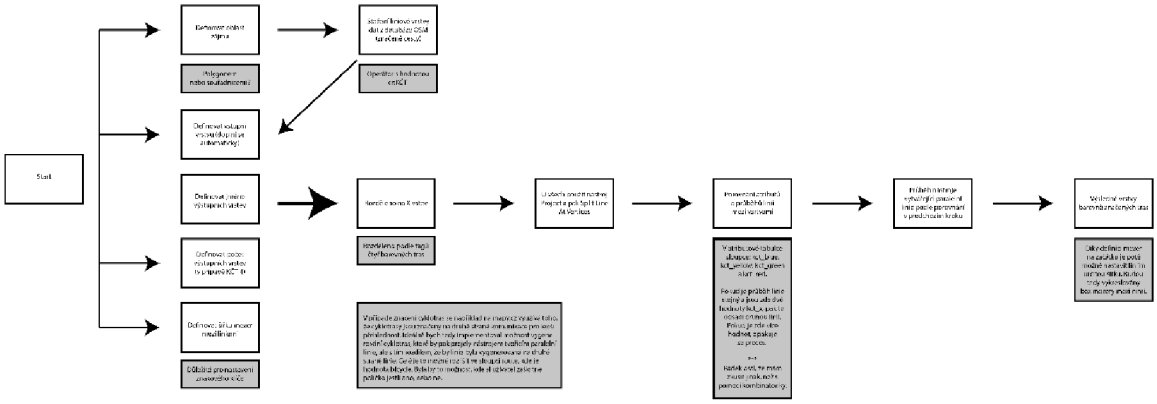
Úvodní odstavec části této práce je věnován motivaci a důvodu vytvoření této práce společně s vymezením kapitol a jejich stručného popisu. Při tvorbě turistických map se autoři setkávají s problematickým zobrazováním značených turistických tras v případě, že podél jedné komunikace vedou dvě a více značených tras. Problém s nastavováním offsetů, tedy odsazení, je popsán v sekci současného stavu řešené problematiky (kapitola 2). V současné době je jednou z možností vizualizace použití kombinatoriky, které s vyšším počtem barevných značených tras nabývá na časové náročnosti. Jednotlivým kombinacím barevných tras vedoucích vedle sebe se musí pro každou jednotlivou variantu nastavovat znakový klíč ručně společně s nastavováním offsetů. Motivací pro vytvoření této práce je tedy zjednodušení procesu tvorby značených tras na mapách. Nutně se nemusí jednat jen o turistické trasy, lze také vizualizovat například i cyklistické, běžecké, nebo ferratové trasy. Tato práce je zaměřena na automatizaci tohoto procesu, a to od detekce barevných tras až po vložení do mapy v prostředí **ArcGIS Pro** s předem nastaveným znakovým klíčem.

Nástroje vyvíjené v této části práce jsou primárně určeny pro data z **OSM**, ale dokáží zpracovat i data z jiných zdrojů po splnění určitých podmínek. Jakmile skript zkontroluje správnost vstupních parametrů jsou vstupním vrstvám definovány posuny na základě určeného pořadí nástrojů. Do nového pole jsou rovněž identifikovány barvy. Následující kroky vedou k samotnému generování značených tras v aktuální mapě v **ArcGIS Pro**. Díky dvěma zvoleným přístupům generování si uživatel může vybrat, dle své preference. S koncem skriptu je do mapy přidána i výstupní vrstva komunikací vybraných linií.

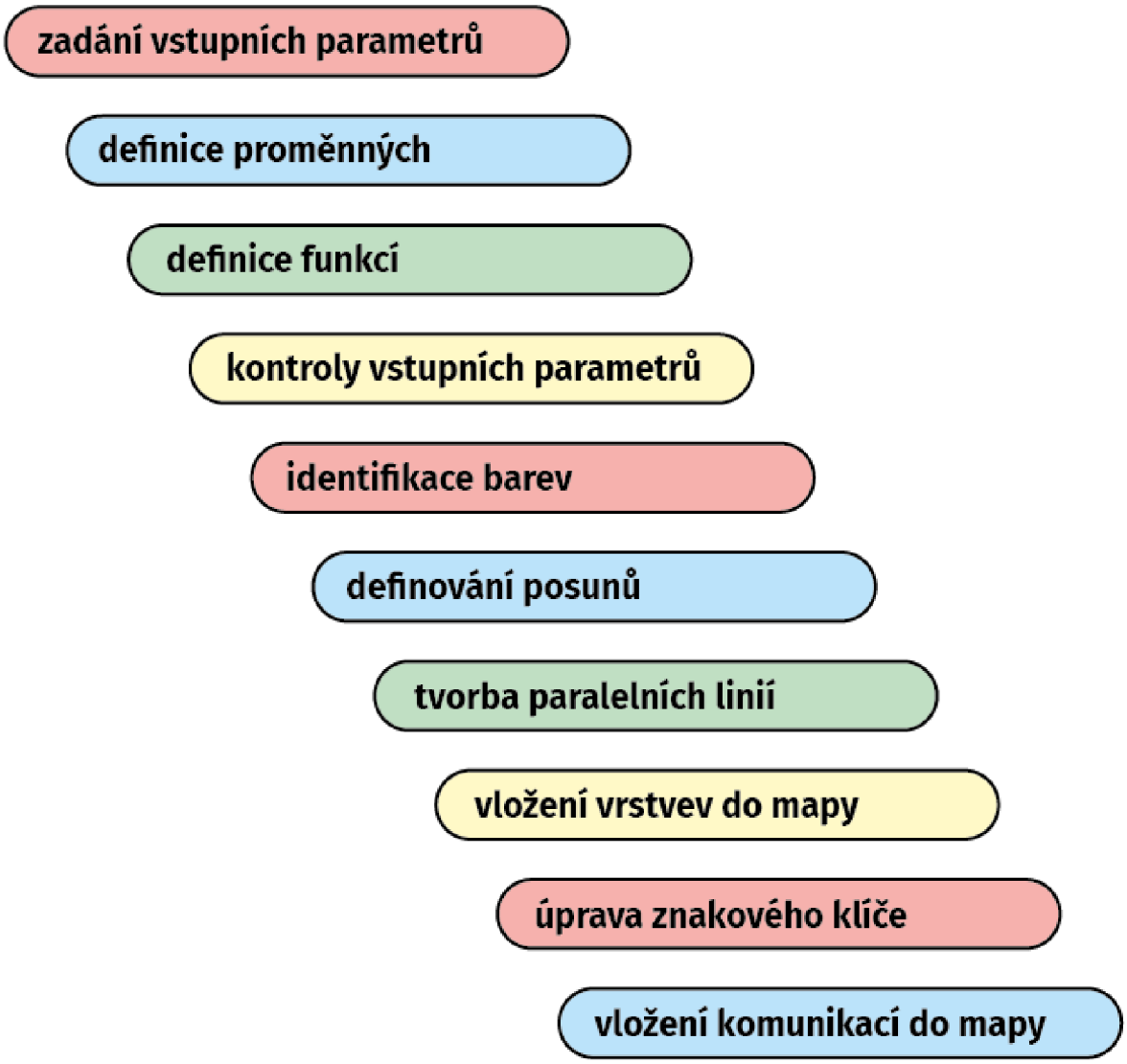
V úvodní kapitole 4.1 je popsán návrh algoritmu s doprovodnými obrázky pro lepší ilustraci a pochopitelnost. Následuje kapitola 4.2, která je věnována tvorbě skriptu v prostředí programu **Visual Studio Code** a popisu vyvíjeného kódu. Kapitola 4.3 je věnována přesunu vyvinutého kódu do prostředí GIS, konkrétněji do **ArcGIS Pro** a v něm tvorba nástrojů a jejich testování. Kapitoly 4.4 a 4.5 jsou pak věnovány dokumentaci v obou případech tvorby skriptů, tedy jak v souboru, tak v nápovědách nástrojů. V poslední kapitole 4.6 je popsána tvorba finální ukázkové mapy, ve které jsou demonstrovány výsledky vyvinutých nástrojů.

4.1 Návrh algoritmu a design skriptu

Před samotným začátkem programování bylo nutné vytvořit návrh algoritmu. Nejdůležitější částí je vymezení postupných kroků, tedy v jakém pořadí budou jednotlivé nástroje spouštěny tak, aby bylo dosaženo výsledné finální podoby dat. Původní návrh algoritmu (obr. 5) byl oproti finální verzi změněn (obr. 6). Změny byly postupně prováděny hlavně z důvodu postupného tvoření skriptu a zjištění alternativních řešení.



Obrázek 5 – Původní návrh algoritmu

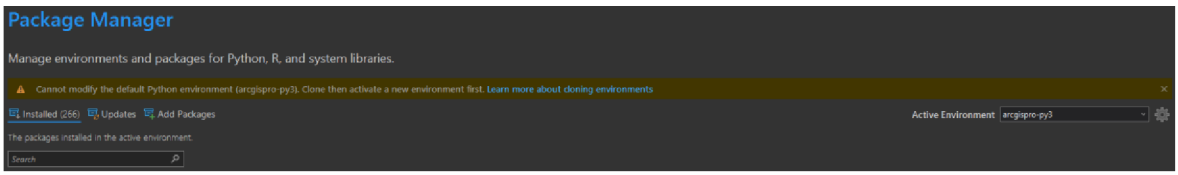


Obrázek 6 – Finální verze návrhu algoritmu

V rámci návrhu algoritmu a designu skriptu bylo rovněž důležité nastudování knihoven programovacího jazyka Python. Původně totiž bylo počítáno s použitím více knihoven. Byly tedy testovány možnosti knihoven, které jsou schopné pracovat s geoprostorovými daty. Mezi testované knihovny se řadí například **Shapely, Fiona, PyProj nebo Geopandas**. Zjišťována byla také vzájemná kompatibilita mezi jednotlivými knihovnami.

S postupným tvořením skriptu vyšlo najevo, že k zpracování výsledných skriptů stačí jen jedna knihovna, a to **ArcPy**. Ta je vyvíjena a její funkcionalita rozšiřována americkou firmou **Esri**, která je zároveň tvůrcem programu **Arcgis Pro**. Jelikož i tento program je úzce spjatý s jazykem Python, je práce s ArcPy v tomto prostředí nejjednodušší. ArcPy nabízí velké množství nástrojů, díky kterým je možné dosažení finálního výsledku.

Dalším, a velice důležitým, důvodem využití jen knihovny ArcPy je importování ostatních knihoven do prostředí ArcGIS Pro. V základním běhovém prostředí (anglicky active environment) je sice mnoho rozšíření, ale z výše zmíněných knihoven není součástí ani jedna z nich, kromě knihovny ArcPy. Pro přidání těchto knihoven do prostředí ArcGIS Pro by bylo nutné zkopírovat a vytvořit nové Python běhové prostředí v rámci programu v sekci Package Manager (obr. 7). Do nově vytvořeného prostředí by se poté daly importovat vybrané knihovny a běhové prostředí by se následně aktivovalo. I přes to se ale jedná o manipulaci se základním běhovým prostředím, což není primárně doporučováno. V případě, že by byl skript používán mimo prostředí ArcGIS Pro, což je také možné, by s importováním knihoven nebyl problém. Pokud by ale uživatel chtěl pracovat i s knihovnou ArcPy, bylo by poté nutné využívat verzi Pythonu, jejíž součástí je i tato knihovna. Uživatel by tedy musel pomocí pip příkazů nové knihovny instalovat do složky v počítači, kde se nachází verze Pythonu určená pro Arcgis Pro. Tento přístup by byl jednodušší oproti kopírování běhového prostředí. V případě, že by uživatel neměl licenci na ArcGIS Pro, a tedy ani na knihovnu ArcPy, by ale skript nefungoval.



Obrázek 7 – Package Manager v ArcGIS Pro

4.2 Tvorba a popis skriptů v programovacím prostředí

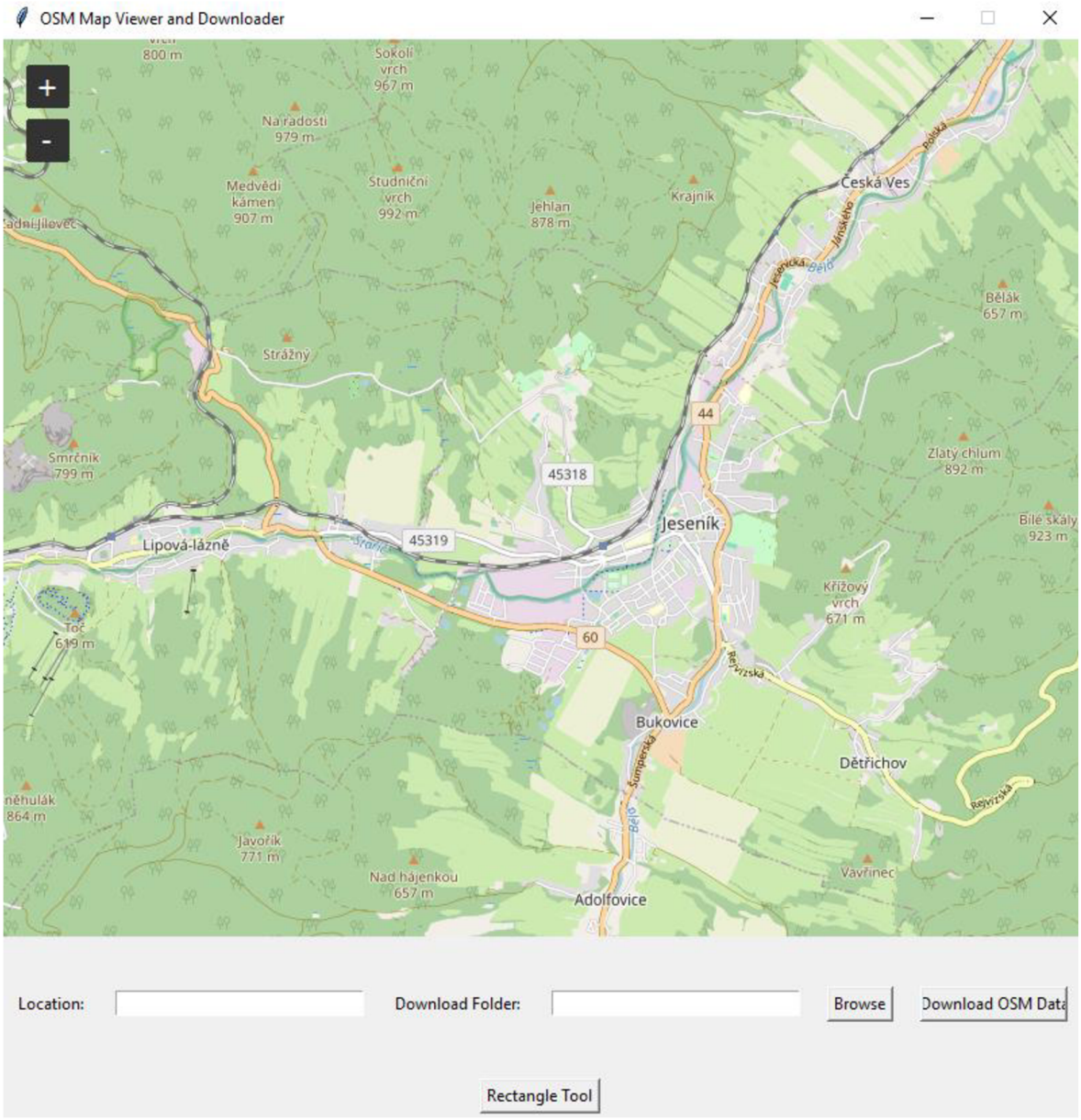
Jakmile byl proveden první návrh algoritmu, začaly práce na tvorbě skriptů. Pro tuto část práce byl využíván program Visual Studio Code. Tento program byl vybrán na základě více kritérií:

- jedná se o tzv. **lightweight editor**,
- podporuje **více programovacích jazyků**,
- je **vysoce přizpůsobitelný** a má **mnoho rozšíření** (extenzí).

Díky tomu, že se jedná o „lightweight“ editor, bylo psaní kódu ulehčeno. Tyto druhy editorů jsou populární právě pro svou jednoduchost, rychlost a podporu více programovacích jazyků. Zároveň nemají velké nároky jak na hardware, tak na software.

V této podkapitole bude tedy popsán postupně kód a jeho dílčí části. Bude popsána funkcionalita jednotlivých sekcí kódu společně s parametry, proměnnými a funkcemi.

V prvotní fázi byla testována tvorba skriptů s grafickým uživatelským prostředím (**GUI**), od kterého bylo nakonec upuštěno. GUI prostředí by umožnilo fungování skriptu i mimo prostředí ArcGIS Pro, ale jen v případě licencované knihovny ArcPy. Prostředí programu ArcGIS Pro bylo zvoleno z více důvodů. Hlavním z nich byla ale vzájemná provázanost s Pythonem a možnost tvorby nástrojů. Na obrázku níže lze vidět zkušební GUI skript pro zobrazování map z OpenStreetMap ajejich následné stažení (obr. 8). Jedním z dalších zkušebních GUI skriptů byl skript na kopírování dat.



Obrázek 8 – GUI okno skriptu s mapami OSM

Po rozhodnutí nevyužívat GUI skripty byly započaty práce na hlavním skriptu, který automaticky vytváří značené turistické trasy v mapě. Skript byl postupně vytvářen ve Visual Studio Code a jeho funkcionalita testována v ArcGIS Pro. V kapitolách níže je kód postupně popsán.

4.2.1 Import knihovny a vstupní parametry

Funkčním začátkem skriptu je importování Python knihoven, které jsou pro správný běh skriptů důležité. Pseudokód, využití a popis parametrů, které jsou na úplném počátku skriptu, jsou popsány v kapitole 4.4. Pro tento skript byla vybrána, a je tedy dostačující, jen jedna Python knihovna, a tou je **ArcPy**, jejíž funkce a nástroje jsou ve finálním nástroji hojně využívány. Na níže uvedeném řádku kódu je knihovna importována společně s částí „as ap“. Tato části zajistí jednodušší psaní kódu a zredukuje počet znaků. Uživatel tak nemusí před každým voláním knihovny psát „arcpy. + nástroj“, ale stačí napsat jen „ap. + nástroj“.

```
import arcpy as ap
```

```
# Setting parameters that will be defined by the user
try:
    ap.env.workspace = ap.GetParameterAsText(0)
    INPUT_LAYER_HIKING = ap.GetParameterAsText(1)
    INPUT_LAYER_BICYCLE = ap.GetParameterAsText(2)
    REFERENCE_SCALE = ap.GetParameterAsText(3)
    LINE_WIDTH = float(ap.GetParameterAsText(4).replace(",", ".")) # Converted to
float
    HIKING_COLOURED_WITH_SYMBOLOGY = ap.GetParameterAsText(5)
    HIKING_COLOURED_WITH_PARALLEL = ap.GetParameterAsText(6)
    BICYCLE_COLOURED_WITH_SYMBOLOGY = ap.GetParameterAsText(7)
    BICYCLE_COLOURED_WITH_PARALLEL = ap.GetParameterAsText(8)
except Exception as ex:
    ap.AddMessage(f"Error while obtaining input parameters: {str(ex)}")
    exit()
```

Po importování knihovny ArcPy následuje definování vstupních parametrů skriptu. Parametry jsou vloženy do bloku „try-except“ pro zajištění, že budou případné chyby ve vstupních parametrech zachyceny. Pokud by nějaká chyba nastala, kód se přesune do sekce „except“ a v prostředí ArcGIS Pro ukáže uživateli chybovou hlášku. V této hlášce je zahrnuta sekce informující uživatele, že došlo k chybě a informace o ní, tedy o který parametr se jedná. Jakmile dojde k vygenerování chybové hlášky, skript se pomocí funkce „**exit()**“ sám zastaví. Je to proto, aby v dalším průběhu skriptu nevznikaly další chyby. V úvodním odstavci této podkapitoly bylo zmíněno prvotní importování knihovny, které je pro celý zbytek kódu důležité. Hned za importováním knihovny totiž používáme první funkci, která je s ní spjatá, a to „**ap.env.workspace**“. Díky této funkci je nastaven tzv. „**workspace**“, neboli pracovní prostor pro výchozí umístění vstupů a výstupů nástrojů geoprocessingu. Následně je použita funkce „**GetParameterAsText**“, která získává zadaný parametr jako textový řetězec. Funkce „**GetParameterAsText**“ jsou využívány ve spojitosti s tvorbou nástrojů právě v prostředí ArcGIS Pro. Uživatel může zadat celkem 9 parametrů, které jsou indexovány od 0 do 8. Prvním výše zmíněným parametrem je právě nastavení pracovního prostoru. Další dva parametry jsou věnovány vstupním vrstvám turistických a cyklistických tras. Parametr „**REFERENCE_SCALE**“ odkazuje na referenční měřítko výsledné mapy, které je v následující části skriptu důležité pro výpočet odsazení paralelních linií. „**LINE_WIDTH**“ odkazuje na šířku linie v mapě. Tento parametr je rovněž důležitý při tvorbě paralelních linií. Ovlivňuje totiž obě možnosti vykreslování značených tras, které jsou popsány v podkapitole 4.2.6. Tento parametr je pak převeden na desetinné číslo pomocí funkce float a v případě špatného zadání uživatelem je nahrazena čárka za desetinnou tečku. Poslední čtyři parametry jsou pak věnovány možným výstupům, které si uživatel může zvolit. Jedná se o dvě možnosti generování paralelních linií, a to s pomocí úpravy znakového klíče a nastavení offsetů v sekci „**Symbology**“ a také úprava fyzického průběhu linií s pomocí funkce popsané v podkapitole 4.2.3.

4.2.2 Definice proměnných

V této sekci kódu, po importování knihovny a definování vstupních parametrů, je přistoupeno k definování proměnných a seznamů, které budou postupně hrát roli v dalších

částech skriptu. Tato část kódu inicializuje proměnné a některým také nastavuje hodnoty, které jsou potřebné pro další operace v rámci skriptu.

```
# Defining variables
unique_colours = {"no_colour"} # A set of colours present in the input dataset
colour_order = ["red", "blue", "green", "yellow", "black", "brown", "orange",
"purple", "white", "gold", "pink", "no_colour"] # Defined colour order
map_colour_order = ["no_colour", "pink", "gold", "white", "purple", "orange",
"brown", "black", "yellow", "green", "blue", "red"] # Reversed colour order
colour_rgb_values = [
    [255, 0, 0, 100],     # Red
    [0, 130, 225, 100],   # Blue
    [0, 128, 0, 100],     # Green
    [255, 255, 0, 100],   # Yellow
    [0, 0, 0, 100],       # Black
    [165, 42, 42, 100],   # Brown
    [255, 165, 0, 100],   # Orange
    [128, 0, 128, 100],   # Purple
    [255, 255, 255, 100],# White
    [255, 215, 0, 100],   # Gold
    [255, 150, 200, 100],# Pink
    [200, 200, 200, 100] # No Colour = grey
] # RGB values for colours in colour_order
colour_rgb_dictionary = dict(zip(colour_order, colour_rgb_values)) # Dictionary
for colours and their RGB values
```

První částí této popisované sekce definování proměnných pro barvy. Níže jsou popsány jednotlivé proměnné a jejich funkce ve skriptu:

- **unique_colours**: jedná se o množinu unikátních barev značených tras ve vstupních datech, defaultní hodnota je „no_colour“ pro případ, že by žádné další barvy nebyly identifikovány,
- **colour_order**: seznam autorem definovaných barev ve specifickém pořadí na základě rešerše,
- **map_colour_order**: seznam barev s obráceným pořadím z důvodu správného pořadí vykreslování v prostředí ArcGIS Pro,
- **colour_rgb_values**: seznam s RGB barevnými hodnotami pro jednotlivé barvy v seznamu **colour_order**, tím je zajištěno, že každá barva má svůj RGB kód,
- **colour_rgb_dictionary**: slovník, který spojuje slovníky **colour_order** a **colour_rgb_values**, spojuje tedy barvy s jejich RGB hodnotami.

Následující část definuje proměnné pro práci se samotným ArcGIS projektem. Proměnná „**aprx**“ slouží k práci s uživatelem aktuálně otevřeným ArcGIS projektem. Díky funkci „**ap.mp.ArcGISProject**“ a její hodnotě „**CURRENT**“ je možné přistupovat právě k aktuálnímu projektu. Dále proměnná „**aprxMap**“ vybírá aktuální mapové okno otevřené uživatelem z důvodu pozdějšího přidávání výsledných vrstev do mapy.

```
aprx = ap.mp.ArcGISProject("CURRENT") # Current project variable
aprxMap = aprx.activeMap
```

Po definování dvou předchozích proměnných jsou definovány další. Níže jsou opět popsány jednotlivě společně s jejich funkcemi ve skriptu:

- **shift_pointer**: slouží k uložení posunů při výpočtu paralelních linií,
- **colour_exists**: proměnná zjišťující, zda ve vstupních datech existuje pole s názvem „colour“ (výchozí hodnota je nastavena jako „False“),
- **osmc_symbo_exists**: proměnná zjišťující, zda ve vstupních datech existuje pole s názvem „osmc_symbo“ (výchozí hodnota je nastavena jako „False“),
- **spatial_reference**: proměnná, která je využívána v případech nastavování souřadnicového systému podle vstupních vrstev, u kterých je optimální souřadnicový systém vypočítán,
- **generate_hiking_paths a generate_bicycle_paths**: proměnné, které slouží na konci skriptu ke vložení vrstvy vybraných komunikací, podle kterých byly výsledné vrstvy generovány,
- **distance**: slouží pro výpočet vzdálenosti posunu (v centimetrech) na základě vstupního parametru referenčního měřítka a šířky čáry,
- **direction_hiking a direction_bicycle**: v případě generování paralelních linií bez použití nastavování offsetů v sekci „**Symbology**“ jsou nastaveny defaultní hodnoty pro generování linií na určité straně komunikace, aby se zamezilo překryvům výsledných vrstev.

```
shift_pointer = {} # Storage of occupied shifts
colour_exists = False
osmc_symbo_exists = False
spatial_reference = False
generate_hiking_paths = False
generate_bicycle_paths = False
if REFERENCE_SCALE: # Variables connected to REFERENCE SCALE
    distance = 0.00035304 * float(REFERENCE_SCALE) * LINE_WIDTH
    aprxMap.referenceScale = float(REFERENCE_SCALE) # Setting the reference scale
of current map
    ap.AddMessage("Reference scale has been set to: " + REFERENCE_SCALE + ".")
    direction_hiking = "left" # Default value for generating hiking trails
alongside roads
    direction_bicycle = "right" # Default value for generating bicycle trails
alongside roads
```

Poslední částí sekce definování proměnných patří dočasným vrstvám, které slouží postupně jako výstupní vrstvy prováděných operací. Vrstvy s těmito názvy jsou ukládány do geodatabáze definované uživatelem v sekci vstupních parametrů. Po proběhnutí skriptu a vložení finálních vrstev do aktuální mapy jsou tyto dočasné vrstvy z geodatabáze smazány s pomocí seznamu „**layers_to_delete**“.

```
# Temporary output feature class variables
TEMP_HIKING = "TEMP_HIKING"
TEMP_HIKING_ROADS = "TEMP_HIKING_ROADS"
TEMP_HIKING_SPLIT = "TEMP_HIKING_SPLIT"
TEMP_HIKING_COUNT = "TEMP_HIKING_COUNT"
TEMP_HIKING_IDENTIFIED = "TEMP_HIKING_IDENTIFIED"
TEMP_HIKING_DISSOLVED = "TEMP_HIKING_DISSOLVED"
TEMP_BICYCLE = "TEMP_BICYCLE"
```

```
TEMP_BICYCLE_ROADS = "TEMP_BICYCLE_ROADS"
TEMP_BICYCLE_COUNT = "TEMP_BICYCLE_COUNT"

# List of Temporary layers to be deleted at the end of the script
layers_to_delete = [
    "TEMP_HIKING",
    "TEMP_HIKING_SPLIT",
    "TEMP_HIKING_COUNT",
    "TEMP_HIKING_IDENTIFIED",
    "TEMP_HIKING_DISSOLVED",
    "TEMP_BICYCLE",
    "TEMP_BICYCLE_COUNT"
]
```

4.2.3 Definice funkcí

Jakmile jsou definovány proměnné a vstupní parametry, je nutné přistoupit k definici jednotlivých funkcí. V případě tohoto skriptu jsou definovány dvě funkce:

- **get_optimal_utm_zone**,
- **create_parallel**.

První zmíněná vypočítá **optimální UTM zónu** pro vstupní vrstvy. Zároveň také určí, na které polokouli se data ze vstupní vrstvy rozkládají z důvodu správného **EPSG** kódu pro změnu souřadnicového systému jak datové vrstvy, tak i mapy. Na základě centrálního bodu v rozsahu dat je vypočítána optimální UTM zóna a **hemisféra** je počítána podle proměnné „**central_point_y**“. Tyto hodnoty jsou poté vráceny jako n-tice, které obsahují číslo UTM zóny a hemisféru.

```
# Definition of functions that defines optimal EPSG code and creates parallel
lines
def get_optimal_utm_zone(extent):
    ''' This function determines the optimal UTM zone and hemisphere for a given
geographical extent, based on its central point, and returns a UTM zone number
and detected hemisphere. '''
    # Determine the central point of the extent
    central_point_x = (extent.XMin + extent.XMax) / 2
    central_point_y = (extent.YMin + extent.YMax) / 2

    # Determine the appropriate UTM zone and hemisphere based on the central
points
    hemisphere = 'N' if central_point_y >= 0 else 'S'
    utm_zone = int((central_point_x + 180) / 6) + 1

    return utm_zone, hemisphere
```

Druhá funkce slouží k vytvoření paralelních linií k liniím vstupním. Její součástí jsou tři vstupní parametry:

- **input_line**,
- **parallel_value**,
- **parallel_direction**.

První parametr je tvořen vstupní linií, které je poté vypočítána nová geometrie na základě druhého parametru, který zjistí, o kolik bude daná linie posunuta. Poslední parametr slouží k určení, na kterou stranu linie se bude nová generovaná odsazovat. Funkce tedy počítá posun pro každý bod vstupní linie na základě určené vzdálenosti a směru posunu. Tato funkce fyzicky mění průběh vstupních linií bez toho, aniž by původní linii zkopírovala. Nejdříve tedy s pomocí metody „**getPart**“ získá části vstupní linie, poté je pro každý bod vypočítán posun na základně parametru „**parallel_value**“ a následně je ošetřena možnost generování na obou stranách linie. Bod je pak fyzicky posunut a přidán do nového pole bodů, že kterých je nakonec vytvořena výsledná paralelní odsazená linie. Tato část kódu je inspirována kódem z článku na webu Esri v sekci technical support (Esri, 2020).

```
def create_parallel(input_line, parallel_value, parallel_direction):
    ''' This function generates a parallel polyline to a given input polyline,
with an offset determined by the Shift_value parameter and in the specified
direction (left or right) '''
    line_part = input_line.getPart(0)
    new_point_array = ap.Array()

    for point in line_part:
        distance = input_line.measureOnLine(point)
        point0 = input_line.positionAlongLine(distance - 0.01).firstPoint
        point1 = input_line.positionAlongLine(distance + 0.01).firstPoint
        difference_x = float(point1.X) - float(point0.X)
        difference_y = float(point1.Y) - float(point0.Y)
        length = (difference_x ** 2 + difference_y ** 2) ** 0.5

        horizontal_move = difference_y * float(parallel_value) / length
        vertical_move = -difference_x * float(parallel_value) / length

        if parallel_direction == "left":
            horizontal_move *= -1
            vertical_move *= -1

        new_point = ap.Point(point.X + horizontal_move, point.Y + vertical_move)
        new_point_array.add(new_point)

    parallel_line = ap.Polyline(ap.Array([new_point_array]))

    return parallel_line
```

4.2.4 Kontroly vstupních parametrů

Tato sekce popisuje postup skriptu po definování funkcí, proměnných a vstupních parametrů. Po těchto úvodních krocích je možné přistoupit k prvním důležitým úkonům. V situaci, že uživatel zadal vstupní parametry je stále potřeba zkontrolovat jejich správnost pro případné pokračování skriptu. To je zajištěno s pomocí podmíněných příkazů **if**. Nejdříve je nutné zkontrolovat, zda uživatel definoval alespoň jednu vstupní a výstupní vrstvu. První sekce kódu zkontroluje, zda je definována alespoň jedna vstupní vrstva. Pokud tomu tak není, uživateli se v prostředí ArcGIS Pro zobrazí chybová hláška společně se zastavením programu pomocí funkce **exit()**. Další dva podmíněné příkazy kontrolují, zda uživatel se vstupní vrstvou definoval alespoň jeden z výstupů. Opět, pokud dojde

k případu, že není definována výstupní vrstva, zobrazí se uživateli hláška s varováním, ale skript v tomto případě pokračuje dále, protože je možné, že alespoň jedna výstupní vrstva bude uživatelem definována pro druhou vstupní vrstvu.

```
# Check if user defined at least one input layer
if not INPUT_LAYER_HIKING and not INPUT_LAYER_BICYCLE:
    ap.AddError("Script has no input layers defined. If you want to create
parallel lines, you must define at least one input layer.")
    exit()

# Check if hiking output layer is defined
if INPUT_LAYER_HIKING and not (HIKING_COLOURED_WITH_SYMBOLOGY or
HIKING_COLOURED_WITH_PARALLEL):
    ap.AddWarning("Script has no output layers defined for hiking input layer.")

# Check if bicycle output is defined
if INPUT_LAYER_BICYCLE and not (BICYCLE_COLOURED_WITH_SYMBOLOGY or
BICYCLE_COLOURED_WITH_PARALLEL):
    ap.AddWarning("Script has no output layers defined for bicycle input layer.")
```

4.2.5 Definování posunů

Jakmile jsou zkontrolovány vstupní a výstupní parametry, může se přejít k hlavnímu úkolu tohoto skriptu, a to tvorbě odsazených linií. Před prvním zavoláním nástroje je opět nutné zkontrolovat pár podmínek., které jsou opět doplněny o chybové a varovné hlášky. Je totiž nutné zkontrolovat, zda je definováno měřítko v případě, že si uživatel vybral generování linií s pomocí funkce **create_parallel**. Kromě toho je kontrolována i možnost, kdyby uživatel omylem zadal stejnou cestu k vstupním a výstupním datům. Tento skript je nastaven tak, aby neupravoval vstupní vrstvu, pouze ji zkopíroval a pracoval s kopií. V případě, že by uživatel chtěl generovat linie oběma způsoby a zadal stejná jména pro výstupní vrstvy, je i tato situace ošetřena varovnou hláškou společně s přejmenováním druhé definované výstupní vrstvy.

```
# Option for generating hiking routes
if INPUT_LAYER_HIKING and (HIKING_COLOURED_WITH_SYMBOLOGY or
HIKING_COLOURED_WITH_PARALLEL):

    # Check if reference scale is defined if user wants to generate hiking routes using
parallel lines
    if HIKING_COLOURED_WITH_PARALLEL and not REFERENCE_SCALE:
        ap.AddError("Reference scale must be set to generate hiking parallel lines.
Shift distances for parallel lines will not be determined. If you want parallel lines,
please set the value in the reference scale box.")
        exit()

    if HIKING_COLOURED_WITH_PARALLEL:
        if HIKING_COLOURED_WITH_PARALLEL == INPUT_LAYER_HIKING or
HIKING_COLOURED_WITH_PARALLEL == INPUT_LAYER_BICYCLE:
            ap.AddWarning("Output hiking trails have the same name as one of input
layers. Output layer name will consist of: _output")
            HIKING_COLOURED_WITH_PARALLEL = HIKING_COLOURED_WITH_PARALLEL + "_output"
```

```
    if HIKING_COLOURED_WITH_SYMBOLOGY:
        if HIKING_COLOURED_WITH_SYMBOLOGY == INPUT_LAYER_HIKING or
HIKING_COLOURED_WITH_SYMBOLOGY == INPUT_LAYER_BICYCLE:
            ap.AddWarning("Output hiking trails have the same name as one of
input layers. Output layer name will consist of: _output")
            HIKING_COLOURED_WITH_SYMBOLOGY = HIKING_COLOURED_WITH_SYMBOLOGY +
"_output"

    if HIKING_COLOURED_WITH_SYMBOLOGY == HIKING_COLOURED_WITH_PARALLEL:
        ap.AddWarning("Both output hiking trails have the same name. Hiking
trails using parallel lines will be named: " +
HIKING_COLOURED_WITH_SYMBOLOGY.split("\\")[-1] + "_parallel.")
        HIKING_COLOURED_WITH_PARALLEL = HIKING_COLOURED_WITH_SYMBOLOGY +
"_parallel"
```

V kódu je následně zakomentována část, která kontroluje, zda je vstupní vrstva složená z linií (bodová či polygonová by nefungovala). Tato část kódu zakomentována z důvodu, že v prostředí ArcGIS Pro je možné nastavit filtr pro vstupní vrstvy, což je popsáno více v podkapitole 4.3. Pokud jsou všechny výše zmíněné podmínky správně splněny, je možno přistoupit k volání prvních nástrojů. Jak již bylo výše zmíněno, skript neupravuje původní data, proto je jako první nástroj z knihovny ArcPy volán „**Project_management**“. Ten zkopíruje původní a zajistí, aby byla první dočasná vrstva ve stupních s pomocí změny souřadnicového systému na **WGS 84** s EPSG kódem 4326. Součástí některých nástrojů jsou rovněž generovány hlášky o úspěšném provedení nástroje, což je více popsáno v podkapitole 4.4. Po zkopírování vrstvy a změny jejího souřadnicového systému je přistoupeno ke krokům, pomocí kterých je zjištěn rozsah vrstvy neboli **extent**. Právě díky rozsahu vrstvy je poté využita funkce **get_optimal_utm_zone** popsána v podkapitole 4.2.3. Po výpočtu optimální UTM zóny a hemisféry je definován EPSG kód pro následné převedení vrstvy do vhodného souřadnicového systému. V případě, že jsou vstupní data na severní polokouli, EPSG kód začíná číslem 32600, ke kterému je poté přičtena hodnota optimální UTM zóny. V případě, že se bude jednat o jižní polokouli, EPSG kód bude začínat číslem 32700. Po vygenerování EPSG kódu je opět využita funkce „**Project_management**“, která převede vrstvu do vhodného souřadnicového systému. Následně je i v samotné mapě změněn souřadnicový systém tak, aby byl shodný s tím optimálně vypočítaným. Na konci této sekce je poté změněna proměnná „**spatial_reference**“ z důvodu, aby tento krok nebyl v případě generování cyklistických nebo běžeckých tras opakován.

```
    Ap.AddMessage("Hiking trails will be part of the final result.")

    # Copy feature so that original file remains untouched using project tool to
make sure layer projection is in degrees
    ap.Project_management(INPUT_LAYER_HIKING, TEMP_HIKING, "4326")
    ap.AddMessage("Input feature class has been copied 38uccessfully and
projection has been set.")

    # Get the extent of the input layer
    describe_hiking = ap.Describe(TEMP_HIKING)
    extent = describe_hiking.extent
```

```
    extent_message = f"Extent of selected layer is: Xmin: {extent.Xmin}, Ymin:
{extent.Ymin}, Xmax: {extent.Xmax}, Ymax: {extent.Ymax}"
    ap.AddMessage(extent_message)

    # Calculate the optimal UTM zone and use Project tool to apply it
    optimal_utm_zone, hemisphere = get_optimal_utm_zone(extent)
    # Create a spatial reference object for the optimal UTM zone and get EPSG
code
    spatial_reference_hiking = ap.SpatialReference()
    spatial_reference_hiking.factoryCode = 32600 + optimal_utm_zone if hemisphere
== 'N' else 32700 + optimal_utm_zone
    # Project the input layer to the optimal UTM zone with determined EPSG code
    ap.Project_management(TEMP_HIKING, TEMP_HIKING_ROADS,
spatial_reference_hiking)

    # Create a Describe object for the layer TEMP_HIKING_ROADS
    describe_project = ap.Describe(TEMP_HIKING_ROADS)
    # Get the EPSG code of projected layer and use it as a new spatial reference
    epsg_code_project = describe_project.spatialReference.factoryCode
    ap.AddMessage(f"Calculated EPSG code for output feature class is:
{epsg_code_project}. Spatial reference will be updated.")
    new_spatial_reference = ap.SpatialReference(epsg_code_project)
    aprxMap.spatialReference = new_spatial_reference
    spatial_reference = True
```

Následující část kódu je věnována polím s barvami jednotlivých turistických tras, jejich identifikacím a převodu hodnot. Nejdříve je s pomocí funkce „**ListFields**“ vytvořen seznam polí pro vstupní vrstvu a následně je zjišťováno, zda se v tomto seznamu nachází 3 důležitá pole pro tento skript:

- **route**,
- **colour**,
- **osmc_symbo**.

V případě, že pole „**route**“ existuje, jsou vybrány jen takové záznamy, které obsahují hodnotu pole buď „**hiking**“ nebo „**foot**“. To je docíleno s pomocí nástroje „**MakeFeatureLayer**“ ze sekce „**management**“, který vytvoří dočasnou vrstvu, ve které vybere všechny ostatní záznamy a ty pak z původní vrstvy smaže. Tím se filtrují data tak, aby se v další části skriptu pracovalo jen s liniemi turistických tras. Jelikož je tento skript ale částečně univerzální, počítá se také s variantou, že vstupní data takovýto atribut mít nebudou. Pokud tato situace nastane, je předpokládáno, že jsou ve vstupních datech všechny linie určené pro turistiku. Dále je kontrolována existence atributů „**colour**“ a „**osmc_symbo**“. V případě, že existují, vrátí se hodnota proměnných jako „**True**“. Po následném přidání nového pole s názvem „**colour_identified**“ s pomocí nástroje „**AddField**“ je přistoupeno k samotné identifikaci barev. Nejdříve je zkontrolováno, jestli existuje alespoň jedno z těchto dvou polí s možnou varovnou hláškou a zadáním hodnot jako „**no_colour**“. K datům se následně skript dostane pomocí funkce „**UpdateCursor**“, který iteruje jednotlivé prvky a může měnit jejich hodnoty v atributové tabulce. Hodnoty z pole „**colour**“ jsou porovnávány s listem „**colour_order**“ obsahujícím barvy a při splnění podmínky je hodnota barvy přepsána do nově vytvořeného pole. Jelikož jsou data z OMS

nedostačující a obsahují právě dva atributy, ve kterých se může nacházet barva značené trasy, je proto nutné zkontrolovat i druhý atribut v datech. Proto je poté přistoupeno, opět pomocí funkce iterující záznamy, k identifikaci barev ještě jednou. Pole „**osmc_symbo**“ je svým obsahem poněkud komplexnější, jelikož neobsahuje jen informaci o barvě, ale o celém znaku v reálném světě. Hodnota tohoto pole může být například „**red:white:red_bar**“, což znamená, že se jedná o červenou páskovou značku s dvěma bílými pruhu jak ji známe třeba v Česku. Z tohoto textového řetězce je vyjmuta jen informace o barvě, tedy první písmena až do dvojtečky s pomocí funkce „**split**“. Pro případ, že po první iteraci je nějaký záznam v nově vytvořeném poli prázdný, nebo má hodnotu „**no_colour**“, je právě prováděna druhá iterace, která z výše zmíněného atributu identifikuje barvy a převede je do nového pole identifikovaných barev. Ty jsou také přidávány do množiny „**unique_colours**“, která je důležitá pro další část skriptu. V těchto iteracích jsou zároveň použity funkce „**lower**“ a „**strip**“, kdy první zmíněná převádí detekované textové řetězce na malá písmena z důvodu, že je potřeba je porovnat s barvami v listu „**colour_order**“. Druhá funkce očistí detekovaný textový řetězec o mezery. A jelikož data z OSM obsahují velké množství atributů, které mohou zpomalit chod skriptu, tak v poslední části této sekce kódu jsou smazány všechny, kromě vybraných čtyř:

- **OBJECTID**,
- **Shape**,
- **Colour_identified**,
- **Shape_Lenght**.

```
    # Create a list of fields and check if "route", "colour" and "osmc_symbo"
fields exists
    fields = ap.ListFields(TEMP_HIKING_ROADS)
    for field in fields:
        if field.name == "route":
            ap.AddMessage("Route field has been succesfully found.")
            # Selecting features based on the where clause and deleting them. It
is done to select only hiking and foot line features
            HIKING_PROJECT_LAYER =
ap.management.MakeFeatureLayer(TEMP_HIKING_ROADS, "HIKING_PROJECT_LAYER", "route
<> 'hiking' AND route <> 'foot'")
            ap.management.DeleteFeatures(HIKING_PROJECT_LAYER)
        if field.name.lower() == "colour":
            colour_exists = True
            ap.AddMessage("Colour field has been succesfully found.")
        if field.name == "osmc_symbo":
            osmc_symbo_exists = True
            ap.AddMessage("Osmc_symbo field has been succesfully found.")

    # Add a field to store colors and determine them from colour or osmc_symbo
    ap.management.AddField(TEMP_HIKING_ROADS, "colour_identified", "TEXT")
    ap.AddMessage("New field for colour values has been created.")
    # If there are no colour and osmc_symbo fields
    if not colour_exists and not osmc_symbo_exists:
        ap.AddWarning ("There are no fields colour or osmc_symbo in attributes of
the input feature class. Colours could not be determined.")
```

```
        with ap.da.UpdateCursor(TEMP_HIKING_ROADS, ["colour_identified"]) as
cursor:
            for row in cursor:
                row[0] = "no_colour"
    # Check colour field first and determine values
    if colour_exists:
        with ap.da.UpdateCursor(TEMP_HIKING_ROADS, ["colour",
"colour_identified"]) as cursor:
            for row in cursor:
                if row[0] is None:
                    row[1] = "no_colour"
                else:
                    if row[0].strip().lower() not in colour_order:
                        row[1] = "no_colour"
                    else:
                        # Add the color to the set
                        unique_colours.add(row[0].lower())
                        # Copy colour to the new field
                        row[1] = row[0].lower()
                cursor.updateRow(row)
    # Secondly, check osmc_symbo field determine missing values
    if osmc_symbo_exists:
        with ap.da.UpdateCursor(TEMP_HIKING_ROADS, ["osmc_symbo",
"colour_identified"]) as cursor:
            for row in cursor:
                if row[0] is None:
                    if row[1] is None:
                        row[1] = "no_colour"
                elif row[0] is not None:
                    # Remove leading and trailing whitespace, split the string
based on ':' and take the first part
                    colour = row[0].strip().split(':')[0].lower()
                    # Add the color to the set
                    unique_colours.add(colour)
                    # Copy colour to the new field
                    if row[1] == "no_colour" and (" " not in row[0] and ":" in
row[0]):
                        row[1] = colour if colour in colour_order else
"no_colour"
                    if row[1] is None:
                        row[1] = colour if colour in colour_order else
"no_colour"
                cursor.updateRow(row)
    ap.AddMessage("Colours have been assigned to the new field and unnecessary
fields will be deleted.")

    # Deleting unnecessary attributes
    for field in ap.ListFields(TEMP_HIKING_ROADS):
```

```
        if field.name not in ["OBJECTID", "Shape", "colour_identified",
"Shape_Length"]:
            ap.management.DeleteField(TEMP_HIKING_ROADS, field.name)
```

Jakmile jsou barvy identifikovány a přebytečné atributy smazány, je možné přistoupit k další části kódu, díky které se dostaneme k definování posunů. Prvním krokem, který je potřeba provést, je zavolání nástroje „**Identity**“. Tento nástroj identifikuje segmenty tras s rozdílnými hodnotami pole „**colour_identified**“. Následně jsou z výsledné vrstvy smazány všechny záznamy, které mají v polích s barvami rozdílné hodnoty. Po smazání segmentů jsou opět smazány nepotřebné atributy. Jakmile jsou data očištěna, dojde ke spuštění nástroje „**CountOverlapingFeatures**“, který zajistí přiřazení jedinečných identifikátorů k určení počtu překrývajících se segmentů. Následně je opět použit nástroj „**Identity**“ k přenesení identifikátorů na segmenty s rozdílnou barvou, ale stejným průběhem. S pomocí nástroje „**Dissolve**“ jsou sloučeny segmenty podle hodnot v polích "**colour_identified**" a "**FID_TEMP_HIKING_COUNT**". Po znovu provedení funkcí „**CountOverlapingFeatures**“ a „**Identity**“ jsou s jistotou identifikovány segmenty a jejich vzájemné překryvy. S následným opětovným odstraněním nepotřebných atributů, přejmenování určitých názvů a přidání nového pole „**Shift**“ je možné pokračovat k další části kódu.

```
# Identify tool to determine the amount of coloured routes on the same path
segment
    ap.analysis.Identity(TEMP_HIKING_ROADS, TEMP_HIKING_ROADS, TEMP_HIKING_SPLIT,
"ALL")
    # Selecting features based on the where clause and deleting them. Deleting
only features that have different values in colour fields
    HIKING_SPLIT_LAYER = ap.management.MakeFeatureLayer(TEMP_HIKING_SPLIT,
"HIKING_SPLIT_LAYER", "colour_identified <> colour_identified_1")
    ap.management.DeleteFeatures(HIKING_SPLIT_LAYER)

    # Deleting unnecessary attributes
    for field in ap.ListFields(TEMP_HIKING_SPLIT):
        if field.name not in ["OBJECTID", "Shape", "colour_identified",
"Shape_Length"]:
            ap.management.DeleteField(TEMP_HIKING_SPLIT, field.name)
    ap.AddMessage("Segments with different colour values have been identified.")

    # Count overlapping features tool in order to generate unique segment IDs
    ap.analysis.CountOverlappingFeatures(TEMP_HIKING_SPLIT, TEMP_HIKING_COUNT)
    # Identify tool in order to transfer segment IDs to segmnents with same line,
but different colour
    ap.analysis.Identity(TEMP_HIKING_SPLIT, TEMP_HIKING_COUNT,
TEMP_HIKING_IDENTIFIED, "ALL")
    ap.AddMessage("Overlapping features have been counted successfully and new
segment IDs have been transferred.")

    # Dissolving features and optimising attributes
```

```
    ap.management.Dissolve(TEMP_HIKING_IDENTIFIED, TEMP_HIKING_DISSOLVED,
["colour_identified", "FID_TEMP_HIKING_COUNT"], [["colour_identified", "FIRST"]],
"MULTI_PART")
    # Count overlapping features 2nd time in order to generate overlapping colour
counts
    ap.analysis.CountOverlappingFeatures(TEMP_HIKING_DISSOLVED,
TEMP_HIKING_COUNT)
    # Identify tool 2nd time in order to transfer colour counts
    ap.analysis.Identity(TEMP_HIKING_DISSOLVED, TEMP_HIKING_COUNT,
TEMP_HIKING_IDENTIFIED, "ALL")
    ap.AddMessage("Features have been dissolved, overlapping features have been
counted successfully and colour counts have been determined. Unnecessary fields
will be deleted.")
    for field in ap.ListFields(TEMP_HIKING_IDENTIFIED):
        if field.name not in ["OBJECTID", "Shape", "FIRST_colour_identified",
"COUNT_", "FID_TEMP_HIKING_COUNT", "Shape_Length"]:
            ap.management.DeleteField(TEMP_HIKING_IDENTIFIED, field.name)
    # Change field names and add new one
    for old_field, new_field in zip(["FIRST_colour_identified", "COUNT_",
"FID_TEMP_HIKING_COUNT"], ["COLOUR", "COUNT", "SEGMENT_ID"]):
        ap.management.AlterField(TEMP_HIKING_IDENTIFIED, old_field, new_field,
clear_field_alias = "CLEAR_ALIAS")
    ap.management.AddField(TEMP_HIKING_IDENTIFIED, "SHIFT", "SHORT")
    ap.AddMessage("Shift field has been added and other fields have been
optimised.")
```

Součástí této části kódu před samotným generováním paralelních tras je definování posunů do nově vytvořeného pole z předchozí části. První iterace s pomocí kurzoru „**UpdateCursor**“ je prováděna pouze pro linie, u kterých je přítomna pouze jedna barva. Pokud k takové situaci dojde, pole „**SHIFT**“ daného prvku následně obdrží hodnotu 1. V případě, že je v rámci jednoho segmentu přítomno více barev je zároveň uložena hodnota do slovníku „**shift_pointer**“ podle identifikátoru „**segment_id**“.

Druhá iterace je prováděna pro segmenty s více barvami. Ta je prováděna nejdříve přes barvy v pořadí určeném proměnnou „**colour_order**“ a následně je kontrolována i přítomnost v proměnné obsahující jedinečné barvy „**unique_colours**“. Pokud jsou tyto podmínky splněny, dochází opět k iteraci s pomocí kurzoru na vybrané vrstvě. Podle pořadí barev jsou jednotlivým segmentům přiřazovány hodnoty do pole „**SHIFT**“. V případě, že je pro dané „**SEGMENT_ID**“ již přiřazena nějaká hodnota, zvýší se hodnota ve slovníku „**shift_pointer**“ o 1 a následujícímu prvku se stejným „**SEGMENT_ID**“ je přiřazena hodnota vyšší. Tímto krokem je dosaženo, že každý prvek se stejným „**SEGMENT_ID**“ bude mít rozdílnou výchozí hodnotu pro budoucí posun. Jako poslední před samotným generováním paralelních linií je přidáno nové pole a do něj hodnoty podle pořadí barev v proměnné „**map_colour_order**“ pro zajištění správného zobrazování v mapách.

```
# Iterating features and defining SHIFT values for single lines
    with ap.da.UpdateCursor(TEMP_HIKING_IDENTIFIED, ["SEGMENT_ID", "COUNT",
"SHIFT"]) as cursor:
        for row in cursor:
            current_count = row[1]
```

```
            segment_id = row[0]

            if current_count == 1:
                row[2] = 1
                cursor.updateRow(row)
            else:
                shift_pointer[segment_id] = 1  # Adds +1 to shift_pointer for
specific segment_id if there are more colours
    # Iterating ordered colours and defining SHIFT for overlapping
lines
    for ordered_colour in colour_order:
        if ordered_colour in unique_colours:
            with ap.da.UpdateCursor(TEMP_HIKING_IDENTIFIED, ["SEGMENT_ID",
"COLOUR", "SHIFT"]) as cursor:
                for row in cursor:
                    segment_id = row[0]
                    colour = row[1]
                    shift = row[2]
                    if not shift and ordered_colour == colour:
                        row[2] = shift_pointer[segment_id]
                        cursor.updateRow(row)
                        shift_pointer[segment_id] += 1
    ap.AddMessage("Shift values have been determined.")

    # Add a new field to store colour order used in the map and determine values
    ap.management.AddField(TEMP_HIKING_IDENTIFIED, "colour_order", "SHORT")
    with ap.da.UpdateCursor(TEMP_HIKING_IDENTIFIED, ["colour", "colour_order"])
as cursor:
        for row in cursor:
            colour = row[0]
            if colour in map_colour_order:
                index = map_colour_order.index(colour) + 1
                row[1] = index
            cursor.updateRow(row)
```

4.2.6 Tvorba paralelních linií

Tato sekce kódu je rozdělená na dvě části. Podle definovaných vstupních parametrů je možné, že budou použity obě sekce kódu. První z nich používá ke generování paralelních linií podél vrstvy komunikací pouze nastavování znakového klíče, kdežto druhá linie fyzicky posouvá a následně jim mění znakový klíč. Linie je tedy možné generovat dvěma různými způsoby generování, které jsou popsány níže.

Pokud je uživatelem ve vstupních parametrech zadána výstupní vrstva pouze s pomocí „**Symbologie**“ v ArcGIS Pro, je proveden kód níže. Ten po převedení linií s pomocí nástroje „**MultipartToSinglepart**“ začne rovnou nastavovat znakový klíč výsledné vrstvy. Do aktuální mapy je přidána tato vrstva a následně je aktualizováno její vykreslování. Podle polí "**colour_order**", "**COLOUR**" a "**SHIFT**". Pro každou třídu je následně aktualizována barva na základě RGB hodnot v „**colour_rgb_dictionary**“. V rámci úprav znakového klíče je změněna i šířka linie, která byla zadána uživatelem v rámci vstupního parametru, a také

popisek v sekci „**Contents**“ v ArcGIS Pro. Hlavním krokem je nicméně nastavení odsazení (offsetů), které jsou vyřešeny právě s pomocí úpravy znakového klíče a linie tedy nejsou posouvány fyzicky, jako je tomu v druhém případě generování paralelních linií. S pomocí funkce „**CreateCIMObjectFromClassName**“ jsou postupně nastaveny offsety jednotlivých tříd na základně hodnot v poli „**SHIFT**“ vynásobenou o hodnotu šířky linie zadanou uživatelem. Základní funkcionalita úpravy znakového klíče s pomocí **ArcPy** bohužel nenabízí úpravu odsazení. Pro tento případ bylo tedy nutné využití funkce „**getDefinition**“. Po poslední aktualizaci znakového klíče jsou uživateli zobrazeny značené turistické trasy ve své finální podobě (obr. 9). Na zmíněném obrázku lze také zpozorovat více stejných hodnot ve vysvětlivkách. To je z důvodu, že například pro každou barvu s hodnotou „**yellow**“ je nastaveno jiné odsazení. V tomto případě došlo, že se žlutá barva generuje na všech pozicích vedle liniového znaku cesty. Na obrázku níže jde vidět, že v určitých místech je žlutá barva generována přímo vedle linie komunikace, kdežto v dalších případech je generována vedle zelené a modré barvy. Viditelná je i varianta, kdy je žlutá barva generována na třetím místě vedle dvojice barev červené a zelené.

```
# Option for generating lines using symbology
    if HIKING_COLOURED_WITH_SYMBOLOGY:
        ap.AddMessage("Hiking trails will be generated using symbology.")
        ap.management.MultipartToSinglepart(TEMP_HIKING_IDENTIFIED,
HIKING_COLOURED_WITH_SYMBOLOGY)
        # SYMBOLOGY PART
        # Add layer to the map
        lyr = aprxMap.addDataFromPath(HIKING_COLOURED_WITH_SYMBOLOGY)
        # Update renderer, RGB values and line width
        sym = lyr.symbology
        sym.updateRenderer('UniqueValueRenderer')
        sym.renderer.fields = ["colour_order", "COLOUR", "SHIFT"]
        for group in sym.renderer.groups:
            for item in group.items:
                colour_labels = item.values
                colour_label = colour_labels[0][1]
                if colour_label in colour_rgb_dictionary:
                    # Assign RGB code from colour_rgb_dictionary
                    rgb_value = colour_rgb_dictionary[colour_label]
                    item.symbol.color = {'RGB': rgb_value}
                else:
                    # Assign default grey color
                    item.symbol.color = {'RGB': [200, 200, 200, 100]}
                # Set the line offset
                item.symbol.size = LINE_WIDTH
        # Set symbology
        lyr.symbology = sym
        # Create new CIM objects to update offsets
        sym = lyr.getDefinition('V3')
        for group in sym.renderer.groups:
            for sym_class in group.classes:
                label = sym_class.label
                shift_value = float(label.split(", ")[2])
                layer_offset = sym_class.symbol.symbol.symbolLayers[0]
```

```
                new_offset =
ap.cim.CreateCIMObjectFromClassName('CIMGeometricEffectOffset', 'V3')
                new_offset.offset = LINE_WIDTH * shift_value
                # Apply new effect
                layer_offset.effects = [new_offset]
                #layer_offset.capStyle = "Butt"
        # Set symbology
        lyr.setDefinition(sym)
        # Change labels to display only colour values
        sym = lyr.symbology
        for group in sym.renderer.groups:
            for item in group.items:
                colour_labels = item.values
                colour_label = colour_labels[0][1]
                item.label = str(colour_label)
        lyr.symbology = sym
```

Obrázek 9 – Značené trasy vygenerované s nastavením offsetů v sekci Symbologie

Druhou možností generování paralelních linií je využití předem definované funkce. V tomto případě ale musí být společně s výstupní vrstvou definováno i referenční měřítko, tedy výsledné měřítko mapy. Jakmile je úspěšně proveden nástroj „**MultipartToSinglepart**“, je přistoupeno k fyzickému posouvání linií s pomocí funkce „**create_parallel**“. Vstupními parametry pro tuto funkci jsou:

- **jednotlivé linie**,
- hodnota pole „**SHIFT**“ vynásobená proměnou „**distance**“,
- **směr generování linií**.

Na základě těchto vstupních parametrů jsou s pomocí funkce a kurzoru linie posouvány. Tento nástroje ale generuje také linie, které protínají samy sebe. Proto je následně v kódu

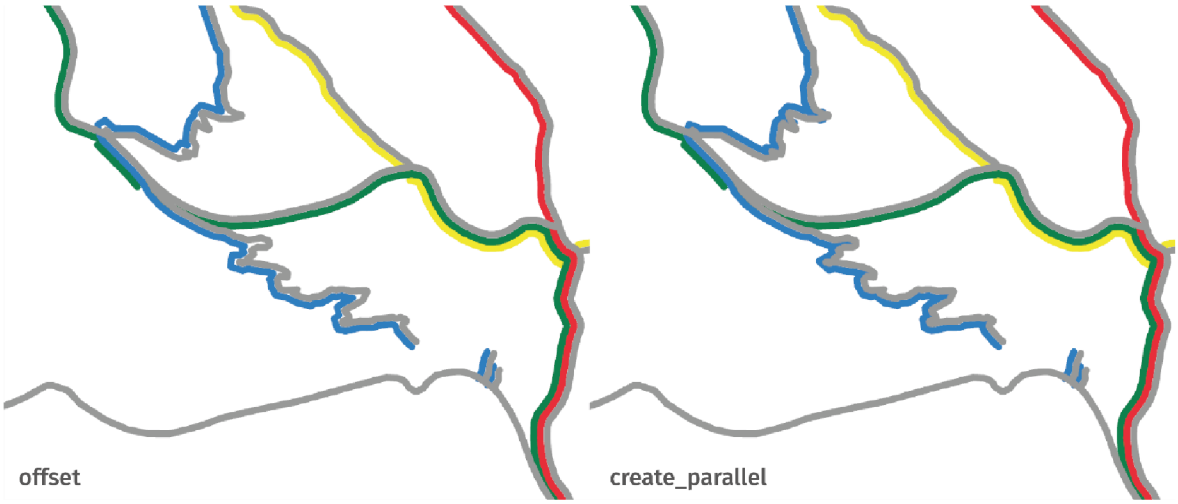
zjištěno, zda má uživatel licencován nástroj „**RepairSelfIntersection**“, který tyto části automaticky opravuje. Pokud není nástroj licencován, je pokračováno v kódu dále. Velké množství vygenerovaných linií, které protínají samy sebe, jsou ale překryty dalšími barevnými liniemi nebo vrstvou komunikací. Licence tohoto nástroje tedy zajistí lepší výsledek skriptu. Následně je už přistoupeno k úpravě znakového klíče. Klíčová změna oproti prvnímu popsanému přístupu je, že jsou linie již odsazeny s pomocí definované funkce a není třeba je odsazovat úpravou znakového klíče. V další sekci kódu jsou tedy upraveny jen barvy linií s pomocí RGB hodnot v „**colour_rgb_dictionary**“, popisky v sekci „**Contents**“ a jejich šířka na základě uživatelem definovaného parametru. Výsledné vysvětlivky jsou pak vygenerovány jen s pomocí čtyř barev (obr. 10). To je zapříčiněno tím, že není potřeba generovat více linií stejné barvy s různým odsazením. Linie se stejnou barvou mají jeden společný atribut, ale jiné odsazení. Díky atributu obsahující barvu je poté možné zobrazit jednoduché vysvětlivky v mapě.

```
# Option for generating lines using parallel lines
    if HIKING_COLOURED_WITH_PARALLEL and REFERENCE_SCALE:
        ap.AddMessage("Hiking trails will be generated using parallel lines.")
        ap.management.MultipartToSinglepart(TEMP_HIKING_IDENTIFIED,
HIKING_COLOURED_WITH_PARALLEL)
        # Shift lines using Create_parallel function
        with ap.da.UpdateCursor(HIKING_COLOURED_WITH_PARALLEL, ("Shape@",
"SHIFT")) as cursor:
            for row, shift_distance in cursor:
                generate_hiking_parallel = create_parallel(row, shift_distance *
distance, direction_hiking)
                cursor.updateRow((generate_hiking_parallel, shift_distance))
        ap.AddMessage("Lines have been been shifted.")
        # Check if Repair Self Intersection tool is licensed. If so, it will be
used to delete self intersections.
        if ap.CheckExtension("Foundation") == "Available":
            ap.topographic.RepairSelfIntersection(HIKING_COLOURED_WITH_PARALLEL,
"DELETE")
            ap.AddMessage("Self-overlaps have been solved.")
        else:
            ap.AddMessage("Repair Self Intersection tool is not licensed. Self-
overlaps have not been solved.")
        # SYMBOLOGY PART
        # Add layer to the map
        lyr = aprxMap.addDataFromPath(HIKING_COLOURED_WITH_PARALLEL)
        # Update renderer, RGB values and line width
        sym = lyr.symbology
        sym.updateRenderer('UniqueValueRenderer')
        sym.renderer.fields = ["colour_order", "COLOUR"]
        for group in sym.renderer.groups:
            for item in group.items:
                colour_labels = item.values
                colour_label = colour_labels[0][1]
                item.label = str(colour_label)
                if item.label in colour_rgb_dictionary:
                    # Assign RGB code from colour_rgb_dictionary
```

```
                rgb_value = colour_rgb_dictionary[item.label]
                item.symbol.color = {'RGB': rgb_value}
                item.symbol.size = LINE_WIDTH
            else:
                # Assign default grey color
                item.symbol.color = {'RGB': [200, 200, 200, 100]}
                item.symbol.size = LINE_WIDTH
    # Set symbology
    lyr.symbology = sym
```

Obrázek 10 – Značené trasy vygenerované s offsety s pomocí funkce create_parallel

Na obrázku níže (obr. 11) také lze vidět rozdíl v generování tras pomocí obou přístupů. V sekcích, kde jsou ostré zatáčky jsou vidět v druhém případě linie protínající samy sebe pod linií komunikací. Naopak linie generovány pomocí změny znakového klíče (nastavením offsetu) jsou odsazovány jiným způsobeb, u kterého jsou viditelné vizuální mezery v některých případech (zejména v úzkých a ostrých zatáčkách).



Obrázek 11 – Rozdíl v generování tras různými přístupy

4.2.7 Generování cyklistických a běžeckých tras

Od této části je skript rozdělen na dva, jelikož v prvním případě se jedná o takzvanou letní a v druhém případě o zimní verzi. U letní verze je možné společně s barevnými trasami generovat i cyklistické stezky na druhé straně linie komunikací. V rámci získání těchto dat jsou vybrány pouze linie, jejichž obsahem je pole „**route**“ s atributy „**bicycle**“ a „**mtb**“. Pro zimní variantu jsou ve skriptu vybírány hodnoty „**ski**“. V případě, že se výše zmíněné pole v atributové tabulce nenachází, je předpokládáno, že jsou všechny linie tvořeny daným tematickým obsahem. Tato část pro generování tras je o poznání jednodušší i díky tomu, že se po splnění podmínek, výpočtu a změně souřadnicového systému a kontrole polí přechází rovnou ke generování paralelních tras. Opět jsou zde dvě možnosti generování, jak bylo již popsáno v podkapitole 4.2.6. Změn oproti turistickým trasám je více. Linie již nejsou generovány s pomocí různých barev a nezáleží na počtu souběžně jdoucích segmentů. Cyklistické a běžecké linie jsou tedy generovány s pomocí jednotného liniového znaku. V kódu níže lze zpozorovat také další změnu, a to přidání nového efektu. „**CIMGeometricEffectDashes**“ přidává linii přerušovanost na základě stanovené šablony. V případě běžeckých tras není tento efekt aplikován z důvodu obecného zobrazení běžeckých tras na mapách. Pro případ generování linií s pomocí funkce „**create_parallel**“ je vytvořen nový atribut „**SHIFT_VALUE**“, díky kterému jsou linie fyzicky posouvány a tím tvořeny paralelní linie.

```
    # Option for generating lines using symbology
    if BICYCLE_COLOURED_WITH_SYMBOLOGY:
        ap.AddMessage("Bicycle trails will be generated using symbology.")
        # Count overlapping features tool used to generate unique segment IDs
        ap.analysis.CountOverlappingFeatures(TEMP_BICYCLE_ROADS,
BICYCLE_COLOURED_WITH_SYMBOLOGY)
        # SYMBOLOGY PART
        # Add layer to the map
        lyr = aprxMap.addDataFromPath(BICYCLE_COLOURED_WITH_SYMBOLOGY)
        # Update renderer, RGB value line width and create new CIM object to
update offset
        sym = lyr.symbology
        sym.updateRenderer('SimpleRenderer')
        lyr.symbology = sym
```

```
        # Turning off labels
        if lyr.supports("SHOWLABELS"):
            # List label classes
            label_classes = lyr.listLabelClasses()
            # Delete expression so that there is no label
            for label_class in label_classes:
                label_class.expression = ""
        # Update RGB value, line width and create new CIM objects to update
offset and apply dash effect
        symbology = lyr.getDefinition('V3')
        symbol = symbology.renderer.symbol.symbol.symbolLayers[0]
        symbol.color = [230, 50, 230]
        new_dash =
ap.cim.CreateCIMObjectFromClassName('CIMGeometricEffectDashes', 'V3')
        new_dash.dashTemplate = [3, 6]
        symbol.width = LINE_WIDTH
        #symbol.capStyle = "Butt"
        new_offset =
ap.cim.CreateCIMObjectFromClassName('CIMGeometricEffectOffset', 'V3')
        new_offset.offset = (-1 * LINE_WIDTH)
        # Apply new effects
        symbol.effects = [new_offset, new_dash]
        # Set symbology
        lyr.setDefinition(symbology)

    # Option for generating lines using parallel lines
    if BICYCLE_COLOURED_WITH_PARALLEL and REFERENCE_SCALE:
        ap.AddMessage("Bicycle trails will be generated using parallel lines.")
        # Count overlapping features in order to generate unique segment IDs
        ap.analysis.CountOverlappingFeatures(TEMP_BICYCLE_ROADS,
TEMP_BICYCLE_COUNT)
        # Multipart to singlepart tool for CopyParallel function to work properly
        ap.management.MultipartToSinglepart(TEMP_BICYCLE_COUNT,
BICYCLE_COLOURED_WITH_PARALLEL)
        # Creating parallel lines for BICYCLE routes.
        ap.AddField_management(BICYCLE_COLOURED_WITH_PARALLEL, "SHIFT_VALUE",
"DOUBLE")
        ap.AddMessage("Overlapping features have been counted successfully and
new field for shift values has been created.")
        # Shift lines using Create_parallel function
        with ap.da.UpdateCursor(BICYCLE_COLOURED_WITH_PARALLEL, ("Shape@",
"SHIFT_VALUE")) as cursor:
            for row, shift_distance in cursor:
                generate_bicycle_parallel = create_parallel(row, distance,
direction_bicycle)
                cursor.updateRow((generate_bicycle_parallel, distance))
        ap.AddMessage("Lines have been been shifted.")
        # Check if Repair Self Intersection tool is licensed. If so, it will be
used to delete self intersections.
```

```
        if ap.CheckExtension("Foundation") == "Available":
            ap.topographic.RepairSelfIntersection(BICYCLE_COLOURED_WITH_PARALLEL,
"DELETE")
            ap.AddMessage("Self-overlaps have been solved.")
        else:
            ap.AddMessage("Repair Self Intersection tool is not licensed. Self-
overlaps have not been solved.")
        # SYMBOLOGY PART
        # Add layer to the map
        lyr = aprxMap.addDataFromPath(BICYCLE_COLOURED_WITH_PARALLEL)
        # Update RGB value, line width and create new CIM object to apply dash
effect
        symbology = lyr.getDefinition('V3')
        symbol = symbology.renderer.symbol.symbol.symbolLayers[0]
        symbol.color = [230, 50, 230]
        new_dash =
ap.cim.CreateCIMObjectFromClassName('CIMGeometricEffectDashes', 'V3')
        new_dash.dashTemplate = [3, 6]
        symbol.width = LINE_WIDTH
        #symbol.capStyle = "Butt"
        # Apply new effect
        symbol.effects = [new_dash]
        # Set symbology
        lyr.setDefinition(symbology)
```

4.2.8 Další kontroly vstupních parametrů

Ve skriptech a nástrojích jsou kontrolovány i další podmínky doprovázené informativními hláškami pro uživatele. Tyto kontroly slouží pro co nejlepší běh skriptů. Jsou zde kombinovány možnosti, kdy by uživatel zapomněl definovat buď vstupní, výstupní nebo obě vrstvy. Pokud by k nějaké podmínce došlo, uživateli se v prostředí **ArcGIS Pro** zobrazí hláška informující právě o nesplnění dané podmínky.

```
elif not (INPUT_LAYER_HIKING and (HIKING_COLOURED_WITH_PARALLEL or
HIKING_COLOURED_WITH_SYMBOLOGY)):
    ap.AddMessage("No hiking routes will be generated.")
    if not INPUT_LAYER_HIKING and (HIKING_COLOURED_WITH_PARALLEL or
HIKING_COLOURED_WITH_SYMBOLOGY):
        ap.AddMessage("If you want hiking routes, you should define input
layer.")
    if INPUT_LAYER_HIKING and not HIKING_COLOURED_WITH_PARALLEL and not
HIKING_COLOURED_WITH_SYMBOLOGY:
        ap.AddMessage("If you want hiking routes, you should define whether you
want to generate hiking routes using symbology or generated parallel lines.")

elif not (INPUT_LAYER_BICYCLE and (BICYCLE_COLOURED_WITH_PARALLEL or
BICYCLE_COLOURED_WITH_SYMBOLOGY)):
    ap.AddMessage("No bicycle routes will be generated.")
    if not INPUT_LAYER_BICYCLE and (BICYCLE_COLOURED_WITH_PARALLEL or
BICYCLE_COLOURED_WITH_SYMBOLOGY):
```

```
        ap.AddMessage("If you want bicycle routes, you should define input
layer.")
    if INPUT_LAYER_BICYCLE and not BICYCLE_COLOURED_WITH_PARALLEL and not
BICYCLE_COLOURED_WITH_SYMBOLOGY:
        ap.AddMessage("If you want bicycle routes, you should define whether you
want to generate hiking routes using symbology or generated parallel lines.")
```

4.2.9 Přidání vrstev komunikací do mapy a zakončení skriptů

Jakmile uživatel vygeneruje alespoň jeden ze čtyř výstupů, bude do finální mapy přidána i vrstva s komunikacemi daného výstupu. Jedná se o tři varianty výstupu:

- přidání vrstvy s turistickými komunikacemi,
- přidání vrstvy s cyklistickými komunikacemi,
- přidání vrstvy s oběma druhy komunikací.

Při splnění stanovených podmínek je pak jedna z vrstev přidána do mapy a následně je jí upraven znakový klíč (barva a šířka linie).

```
# Add paths to the map
if generate_hiking_paths and not generate_bicycle_paths:
    # Add hiking paths to the map
    roads_hiking_path = ap.GetParameterAsText(0) + "\\" + TEMP_HIKING_ROADS
    roads_hiking = aprxMap.addDataFromPath(roads_hiking_path)
    # Update symbology of hiking paths
    sym = roads_hiking.getDefinition('V3')
    symbol = sym.renderer.symbol.symbol.symbolLayers[0]
    symbol.color = [153, 153, 153]
    symbol.width = LINE_WIDTH
    roads_hiking.setDefinition(sym)
    ap.AddMessage("Hiking paths feature class has been added to the map.")

if not generate_hiking_paths and generate_bicycle_paths:
    # Add bicycle paths to the map
    roads_bicycle_path = ap.GetParameterAsText(0) + "\\" + TEMP_BICYCLE_ROADS
    roads_bicyle = aprxMap.addDataFromPath(roads_bicycle_path)
    #Update symbology of bicycle paths
    sym = roads_bicyle.getDefinition('V3')
    symbol = sym.renderer.symbol.symbol.symbolLayers[0]
    symbol.color = [153, 153, 153]
    symbol.width = LINE_WIDTH
    roads_bicyle.setDefinition(sym)
    ap.AddMessage("Bicycle paths feature class has been added to the map.")

if generate_hiking_paths and generate_bicycle_paths:
    ap.management.Merge([TEMP_HIKING_ROADS, TEMP_BICYCLE_ROADS],
"HIKING_BICYCLE_PATHS")
    # Add hiking paths to the map
    roads_hiking_bicycle_path = ap.GetParameterAsText(0) + "\\" +
"HIKING_BICYCLE_PATHS"
    roads_hiking_bicycle = aprxMap.addDataFromPath(roads_hiking_bicycle_path)
    # Update symbology of hiking paths
```

```
    sym = roads_hiking_bicycle.getDefinition('V3')
    symbol = sym.renderer.symbol.symbol.symbolLayers[0]
    symbol.color = [153, 153, 153]
    symbol.width = LINE_WIDTH
    roads_hiking_bicycle.setDefinition(sym)
    ap.AddMessage("Hiking paths were merged with bicycle paths and resulting
feature class has been added to the map.")
```

Poslední řádky kódu jsou poté věnovány smazání dočasných vrstev vytvořených v rámci běhu skriptů. S pomocí nástroje „**Delete**“ jsou vrstvy smazány. Následně je zakomentována možnost uložit projekt, kterou si uživatel může povolit a na konci skriptu je uživateli vyobrazena hláška, že skript proběhl úspěšně.

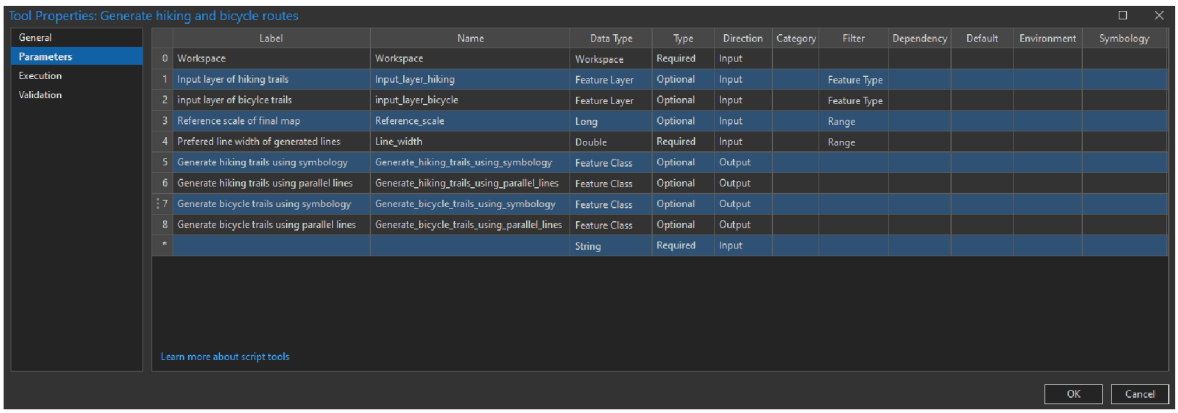
```
# Delete temporary layers
for layer_name in layers_to_delete:
    ap.management.Delete(layer_name, "FeatureClass")
ap.AddMessage("Temporary layers have been deleted.")

# Save changes in project
#aprx.save()
ap.AddMessage("Script has run successfully.")
```

4.3 Tvorba nástrojů v GIS prostředí

Společně s vývojem skriptů v prostředí programu **Visual Studio Code** byl využíván i **ArcGIS Pro**. Pomocí vytváření nových takzvaných „**toolboxů**“, tedy sad nástrojů, je možné vyvíjené skripty importovat do ArcGIS Pro. Toolboxy lze najít v sekci „**Catalog**“, kde se mimo jiné nachází i databáze, mapy nebo třeba přístupy ke složkám. Skripty tedy byly do programu přidány a dále musely být nastaveny tak, aby fungovaly správně.

K správnému nastavení skriptů je využíváno okno, ve kterém se dají nastavit vstupní a výstupní parametry společně s jejich vlastnostmi (obr. 12). V podkapitole 4.2.1 byly tyto parametry rovněž popsán. Nicméně v **ArcGIS Pro** je nutné parametrům nastavit určité vlastnosti a filtry pro správné fungování. Na obrázku níže lze vidět, že bylo nastavováno devět parametrů. Tabulka 2 zobrazuje jména parametrů společně s nastavovanými filtry a vlastnostmi. Sekce „**Name**“ je pouze k pojmenování vstupního parametru. Následující „**Data Type**“ je využíván k nastavení datového typu parametrů, které může uživatel zadat. Vstupními vrstvami tedy mohou být například „**Feature layer**“. Zde se jedná o vrstvy buď v mapě, geodatabázi, nebo samotném PC (například ve formě SHP). Pro nastavení šířky linie byl pak použit datový typ „**Double**“ z důvodu, kdyby uživatel chtěl generovat linie například s tloušťkou obsahující hodnotu za desetinnou čárkou. Pro zadání měřítka je poté dostačující datový typ „**Long**“, jehož součástí mohou být jen celá čísla. Sekce „**Type**“ je určena k nastavení, který parametr je povinný a musí být zadán uživatelem, a který je dobrovolný. Bez nezadaných povinných parametrů nástroj uživateli nedovolí nástroj spustit. Dále s pomocí „**Direction**“ je nastavováno, zda uživatelem zadané vstupy poslouží jako vstupní, nebo výstupní vrstvy. V případě jiných skriptů s nastavením „**Output**“ jsou pro skončení skriptů výsledné vrstvy přidány do mapy. V toto případě jsou ale vrstvy přidávány do mapy v rámci běhu nástroje, a to z důvodu změny znakového klíče. Poslední důležitou součástí nastavování vstupních parametrů je část „**Filter**“, díky které je možné filtrovat vstupní hodnoty parametrů a nedovolit uživateli zadat nesprávná data. Filtry byly použity u čtyř vstupních parametrů. Dvěma vstupním vrstvám bylo nastaveno, že vrstva,

kterou uživatel nahraje do nástroje, může být pouze „**Polyline**“, tedy liniová. Díky tomuto filtru není nutné kontrolovat vstupní data pomocí kódu a ve skriptech je tato část zakomentována. Druhým důležitým filtrem bylo nastavení rozsahu hodnot u zadávání šířky linie a referenčního měřítka. Tyto filtry byly nastavovány z důvodu, aby uživatel například omylem nezadal mínusové hodnoty pro tyto vstupní parametry. „**Reference_scale**“ má tedy nastavenou minimální hodnotu 1 a u „**Line_width**“ je nastavená minimální hodnota 0,1.

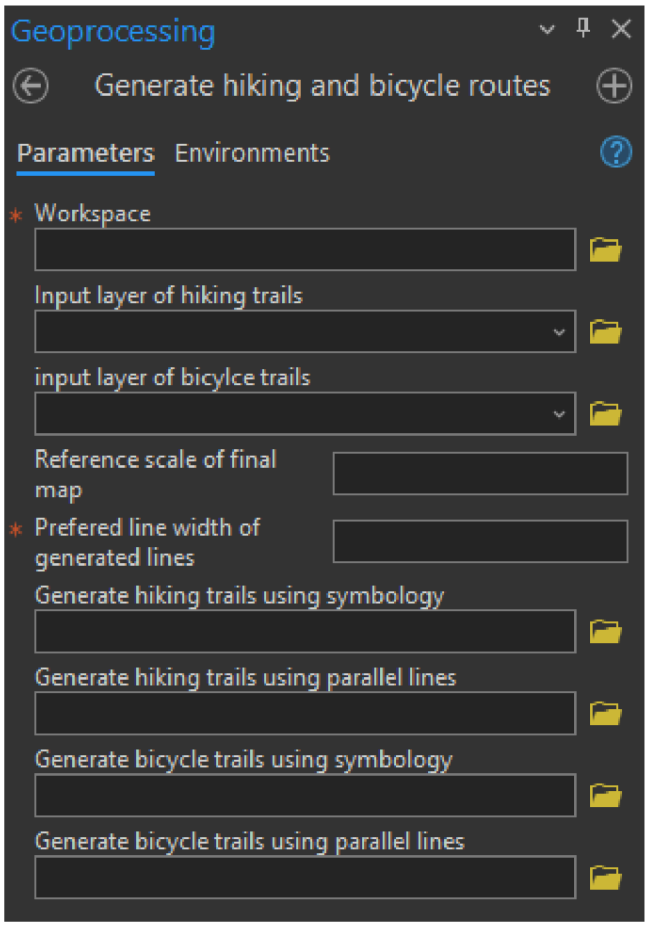


Obrázek 12 – Nastavení parametrů nástroje

Tab. 2: Nastavení vstupních parametrů nástroje v programu ArcGIS Pro

Name	**Data Type**	**Type**	**Direction**	**Filter**
Workspace	Workspace	Required	Input	
Input_layer_hiking	Feature layer	Optional	Input	Feature type
input_layer_bicycle	Feature layer	Optional	Input	Feature type
Reference_scale	Long	Optional	Input	Range
Line_width	Double	Required	Input	Range
Generate_hiking_trails_using_symbology	Feature class	Optional	Output	
Generate_hiking_trails_using_parallel_lines	Feature class	Optional	Output	
Generate_bicycle_trails_using_symbology	Feature class	Optional	Output	
Generate_bicycle_trails_using_parallel_lines	Feature class	Optional	Output	

Jakmile byly skripty nahrány a vlastnosti vstupních parametrů nastaveny, je možné nahlédnout do prostředí samotných nástrojů. V sekci „**Geoprocessing**“, kde se nacházejí nástroje, je možné skript spustit. Na obrázku níže (obr. 13) lze vidět všechny vstupní parametry, které byly nastavovány. V tento moment bylo možné začít s testováním vytvořených nástrojů. S pozdější fází vývoje byly skriptům přidány i nápovědy s dokumentacemi, což je popsáno v podkapitole 4.4.



Obrázek 13 – Nástroj Generate hiking and bicycle routes se vstupními parametry

4.4 Dokumentace

Nedílnou součástí každého skriptu a nástroje je dokumentace společně s nápovědami a vysvětlivkami. Proto jsou součástí skriptů komentáře vysvětlující postupný chod kódu. Šedou barvou zbarvené komentáře na ukázkovém kódu níže uživatele ve vývojovém prostředí skriptu informují o jednotlivých krocích a funkcionalitě.

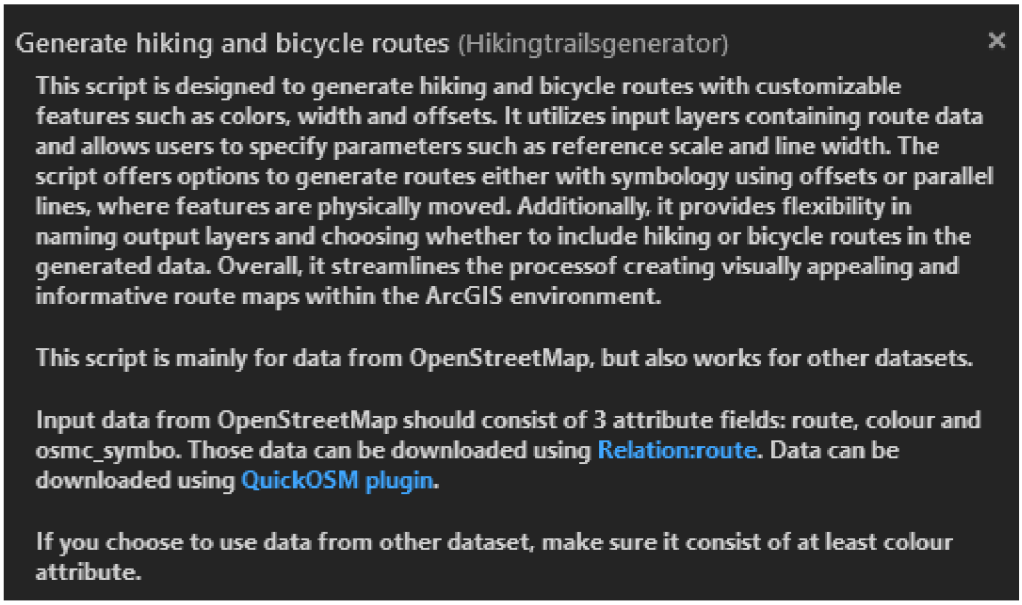
```
# Calculate the optimal UTM zone and use Project tool to apply it
    optimal_utm_zone, hemisphere = get_optimal_utm_zone(extent)
    # Create a spatial reference object for the optimal UTM zone and get EPSG
code
    spatial_reference_hiking = ap.SpatialReference()
    spatial_reference_hiking.factoryCode = 32600 + optimal_utm_zone if hemisphere
== 'N' else 32700 + optimal_utm_zone
    # Project the input layer to the optimal UTM zone with determined EPSG code
    ap.Project_management(TEMP_HIKING, TEMP_HIKING_ROADS,
spatial_reference_hiking)
```

Rovněž je zvykem vložení dokumentace na začátek skriptů (obr. 14). Tato počáteční část poskytuje uživateli základní informace o skriptu společně s pseudokódem (ve kterém je popsáno stručně, jak skript funguje), možnostech využití a v tomto případě i nastavení parametrů v programu **ArcGIS Pro**. Parametry jsou popsány z důvodu, kdyby se k uživateli někdy dostal jen kód.

```
#-------------------------------------------------------------------------------------------------------------------
# Name:          Automation of input data representing marked routes in maps
# Purpose:       Generating coloured hiking routes along lines
# Attachement:   X
#
# Author:        Jakub Pospíšil
# University:    Palacký University Olomouc
# Purpose:       Diploma thesis
#
# Script v.:     1.5
# Created:       08.05.2024
# Licence:       CC BY 4.0
# Licence text:  Automation of input data representing marked routes in maps © 2024 by Bc. Jakub Pospíšil is licensed under CC BY 4.0
#-------------------------------------------------------------------------------------------------------------------

# Pseudocode:
# This script is designed to generate hiking and bicycle routes with customizable features such as colors, width
# and offsets. It utilizes input layers containing route data and allows users to specify parameters such as
# reference scale and line width. # The script offers options to generate routes either with symbology using
# offsets or parallel lines, where features are physically moved. Additionally, it provides flexibility in naming
# output layers and choosing whether to include hiking or bicycle routes in the generated data. Overall, it
# streamlines the processof creating visually appealing and informative route maps within the ArcGIS environment.

# Usage:
# This script is mainly for data from OpenStreetMap, but also works for other datasets. Input data from
# OpenStreetMap should consist of 3 attribute fields: route, colour and osmc_symbo. Those data can be downloaded
# using Relation:route. Data can be downloaded by QuickOSM plugin in QGIS using Key = route. If you choose to use
# data from other dataset, make sure it consist of at least colour attribute.

# Parameters                                                    DATA TYPE       TYPE        DIRECTION   FILTER

# ap.env.workspace: Workspace environment variable for the directory of output, and in some cases also input data.   Workspace       REQUIRED    Input       No filter
# INPUT_LAYER_HIKING: Path to the input layer containing hiking trail features.                                      Feature Layer   OPTIONAL    Input       Feature type - Polyline
# INPUT_LAYER_BICYCLE: Path to the input layer containing bicycle trail features.                                    Feature Layer   OPTIONAL    Input       Feature type - Polyline
# REFERENCE_SCALE: Reference scale affecting the resulting shift values of generated trails using parallel lines.   Long            OPTIONAL    Input       Range - Minimum - 1      - Maximum - 100000000
# LINE_WIDTH: Width of the trails line symbols.                                                                      Double          REQUIRED    Input       Range - Minimum - 0,1    - Maximum - 1000
# HIKING_COLOURED_WITH_SYMBOLOGY: Output name for hiking trails generated using offsets in symbology.               Feature Class   OPTIONAL    Output      No filter
# HIKING_COLOURED_WITH_PARALLEL: Output name for hiking trails generated using paralle lines.                       Feature Class   OPTIONAL    Output      No filter
# BICYCLE_COLOURED_WITH_SYMBOLOGY: Output name for bicycle trails generated using offsets in symbology.             Feature Class   OPTIONAL    Output      No filter
# BICYCLE_COLOURED_WITH_PARALLEL: Output name for bicycle trails generated using parallel lines.                    Feature Class   OPTIONAL    Output      No filter
```

Obrázek 14 – Základní informace, pseudokód, využití a nastavení parametrů

Nejdůležitější částí této sekce byla ale úprava a vložení nápověd pro nástroje v **ArcGIS Pro**. Jakmile se totiž uživatel dostane do situace, kdy si není jistý, jak skript funguje nebo potřebuje poradit se vstupním parametrem, může myší najet na kulaté tlačítko s otazníkem a následně mu budou zobrazovány nápovědy jak už k jednotlivým parametrům, tak k celému skriptu. Na obrázku níže lze například vidět nápovědu, která se uživateli zobrazí po najetí myší na hlavní tlačítko nápovědy k celému nástroji (obr. 15). Všechny nápovědy společně s popisem nástroje a dalšími informacemi je možné si zobrazit také po kliknutí pravým tlačítkem na nástroj a následným kliknutím na tlačítko „**View Metadata**“. Nápovědy a metadata jsou rovněž doplněny o ilustrační obrázky pro lepší pochopení funkcionality nástrojů.

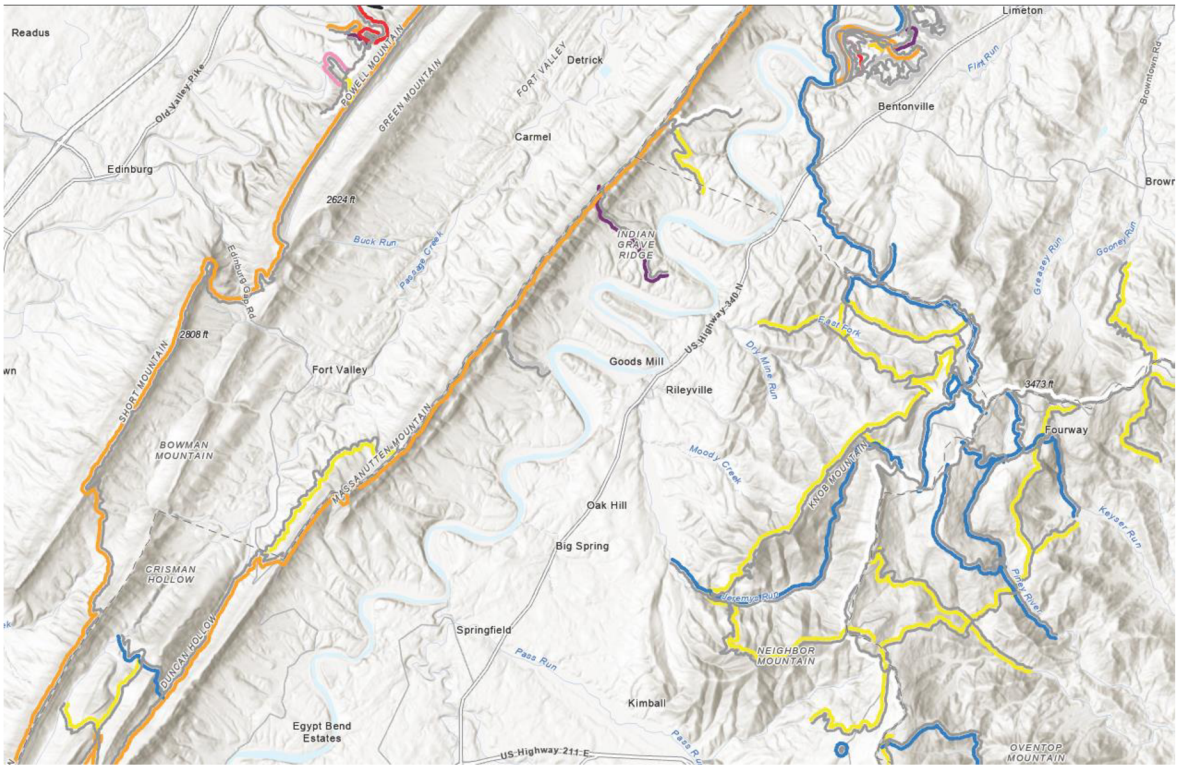


Obrázek 15 – Hlavní nápověda k nástroji Generate hiking and bicycle routes

4.5 Testování

S postupným vývojem skriptů byla testována i jejich funkcionalita. Bylo rovněž nutné nejdříve správně nastavit nástroje v **ArcGIS Pro**. Jakmile byly správně nastaveny, bylo možné začít testovat skripty na zkušebních datech. Postupovalo se od jednoduchých dat ke komplexnějším. Se zobrazujícími se chybovými hláškami byl kód postupně optimalizován a upravován. Se studováním dokumentací a novými nápady byly skripty rovněž vylepšovány a jejich funkcionalita rozšiřována.

Testování bylo nejdříve zaměřeno na různé části kódu a postupně byl testován i celý skript. Byla například stahována data z různých oblastí světa za účelem otestování správného určení UTM zóny, a tedy i EPSG kódu. Data s informacemi o značených trasách byla rovněž testována. Jak již bylo zmíněno v rešerši této práce, v různých oblastech světa jsou barevné trasy značeny různými způsoby a barvami. Proto bylo provedeno testování například i na datech z oblastí národního parku Shenandoah (obr. 16), Andalusie, Jihoafrické republiky, Tenerife, Alp a dalších.

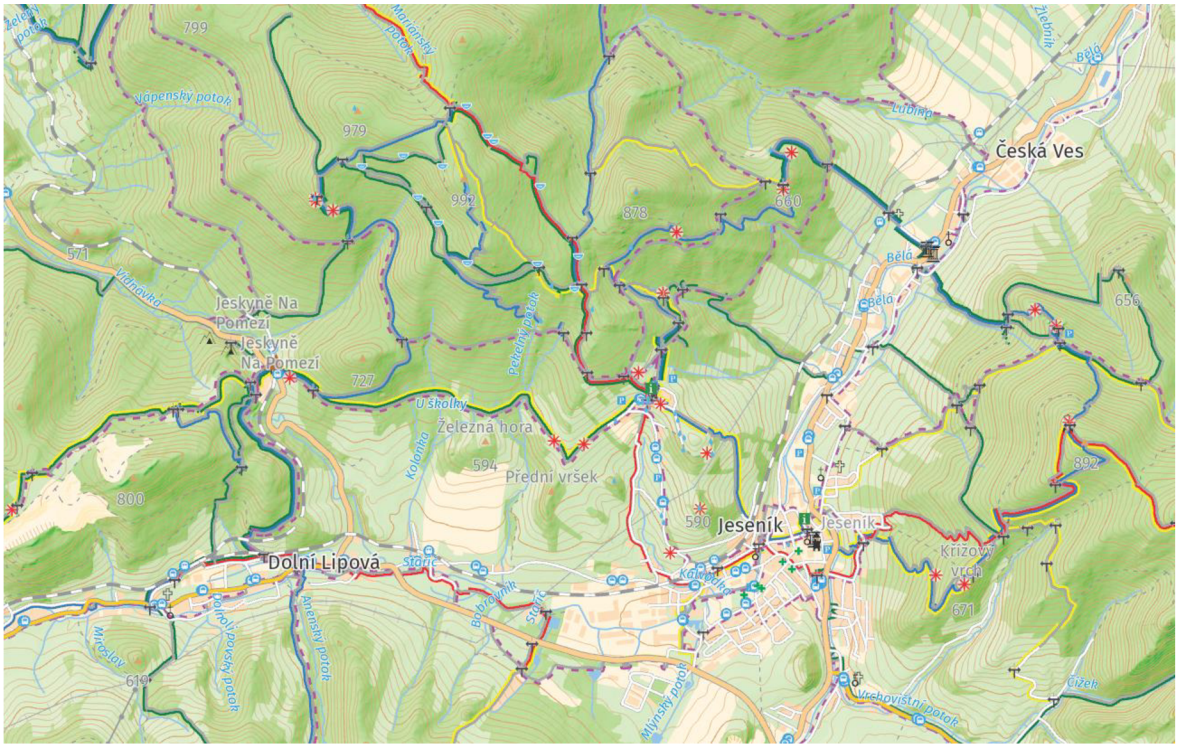


Obrázek 16 – Značené trasy v národním parku Shenandoah

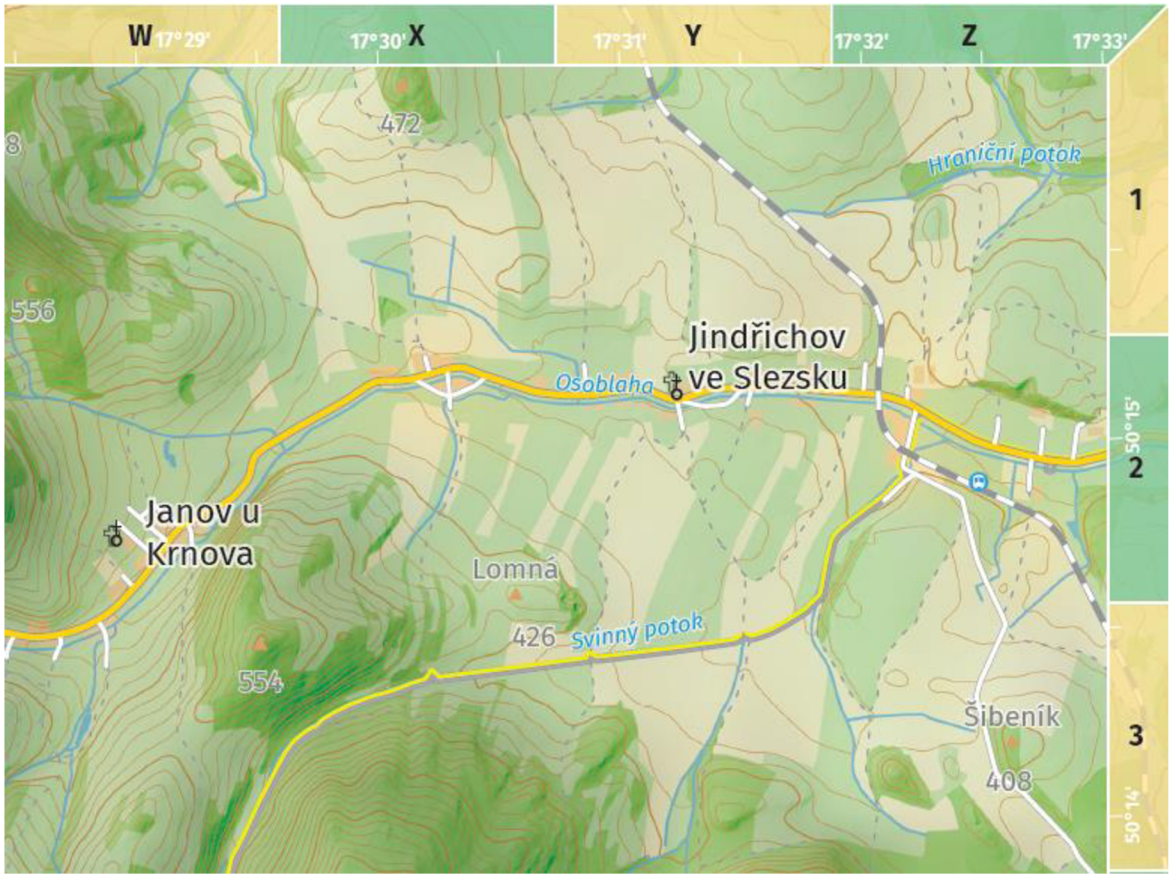
4.6 Tvorba ukázkové mapy

Během finalizace skriptů byla započata i práce na ukázkové mapě, která demonstruje funkcionalitu vyvinutých nástrojů. Byla vybrána oblast Hrubého Jeseníku v měřítku 1 : 40 000. Pro demonstrování funkcionality i druhého vyvinutého nástroje, který je popsán v kapitole 5, byly vybrány dva datové zdroje. Prvním zdrojem je datová sada Data50, ze které byl primárně tvořen podklad mapy. Součástí této sady je i liniová vrstva obsahující lesní cesty, které jsou pro tvorbu turistických map nezbytné. Jako další datový zdroj sloužila data stažena z databáze OSM. Primárně byla stažena liniová vrstva obsahující informace o turistických a cyklistických značených trasách. Z databáze OSM byly rovněž staženy další zájmové body, které doplňují tematickou část mapy. Mezi takové

body se řadí například zastávky, infocentra, vyhlídky, rozcestníky, prameny, parkoviště a další. V rámci tvorby podkladu byl vrstvám postupně upraven znakový klíč. Následně bylo přistoupeno k tvorbě značených tras. Prvním krokem bylo využití nástroje na přichycování linií z OSM na linie z databáze Data50. Jakmile byly linie s atributy přichyceny, přišel na řadu hlavní nástroj této diplomové práce – generování barevných turistických tras. Na obrázku níže lze vidět vygenerované trasy ve výřezu mapového okna spolu s podkladem a dalšími zájmovými body (obr. 17). V rámci studie současně prodávaných turistických map byl vytvořen finální layout, který obsahuje základní prvky – název mapy, legendu, měřítko a tiráž. Do mapy bylo přidáno i naznačení souřadnicové sítě spolu s referenční mřížkou pro lepší orientaci v mapě (obr. 18). Výsledná mapa s rozměrem 90 × 67,5 cm (poměr 4:3) byla vyexportována a finální úpravy byly následně prováděny v programu **Adobe Illustrator**. Mapa byla vytisknuta a tvoří volnou přílohu číslo 4 této diplomové práce.



Obrázek 17 – Výřez mapového okna v ArcGIS Pro



Obrázek 18 – Naznačená souřadnicová síť a referenční mřížka

5 VÝVOJ NÁSTROJE PRO PŘICHYTÁVÁNÍ LINIÍ

Druhá část této diplomové práce byla věnována vývoji nástroje, který dokáže přichytit (**snapnout**) liniové vrstvy z různých datových zdrojů na sebe. Důvodů pro tento krok může být mnoho. Pro účel této práce byl nástroj vyvinut z důvodu návaznosti na hlavní nástroj vytvářející paralelní linie turistických značených tras. Jakmile má tedy uživatel dvě liniové vrstvy a nastane případ, že jsou od sebe linie vzdáleny do určité mezní vzdálenosti, linie do uživatelem vybrané vzdálenosti budou přichyceny na linie vstupní vrstvy. V případě překročení mezní vzdálenosti zůstávají linie nedotčené.

V této části práce byly vyvíjeny dva nástroje, kdy jeden z nich je určen pro úpravu dat z OSM a druhý byl zvolen jako univerzální. Skript, který je určen pro data z OSM, je spjatý s hlavními nástroji vyvíjenými v kapitole 4. Připravuje totiž data pro generování značených tras, které mají jiný průběh než cílená liniová vrstva. Druhý nástroj je více univerzální. Je z něj odstraněna část kódu, která upravuje data z OSM a výsledkem jsou jen přichycené linie společně s přenesenými atributy.

V úvodní kapitole 5.1 je popsána tvorba skriptů v prostředí **Visual Studio Code**, ze kterého byly později skripty převedeny do **ArcGIS Pro** (kapitola 5.2). Stejně tak, jako tomu bylo u vývoje hlavních skriptů, byla tvořena dokumentace, nápovědy a komentáře, které jsou popsány v kapitole 5.3. Poslední kapitola 5.4 je věnována testování skriptů jak samotných, tak ve spojitosti s hlavními skripty generujícími barevné trasy. Kapitoly nejsou již tolik obsáhlé jako v případě čtvrté kapitoly této práce. A to z důvodu, že by se mnoho věcí opakovalo.

5.1 Tvorba skriptu

S myšlenkou v zadání této práce k vytvoření nástroje, který bude řešit generování tras vedle linií s jiným průběhem, byly započaty práce na vývoji dalšího skriptu. Kód byl opět psán v prostředí programu **Visual Studio Code**. V následujících podkapitolách je vývoj skriptu popsán. V rámci následujících podkapitol budou společně popsány oba vyvinuté nástroje společně.

5.1.1 Import knihovny, parametry a definování proměnných

Úvodní sekce kódu je věnována importu knihovny **ArcPy**, definování vstupních parametrů a proměnných. Kód obsahuje jen pět vstupních parametrů, ve kterých uživatel zadá pracovní prostředí, dvě vstupní liniové vrstvy, jednu výstupní vrstvu a takzvaný „**Threshold**“, neboli práh či mezní hodnotu. Následně jsou definovány dočasné vrstvy, které slouží jako výstupy jednotlivých nástrojů z knihovny **ArcPy**. Poslední částí kódu v této sekci je list dočasných vrstev, díky kterému jsou na konci skriptu dočasné vrstvy smazány.

```
import arcpy as ap

# Setting parameters that can be defined by user
ap.env.workspace = ap.GetParameterAsText(0)
INPUT_ROADS = ap.GetParameterAsText(1)
INPUT_TOURIST = ap.GetParameterAsText(2)
OUTPUT_TOURIST = ap.GetParameterAsText(3)
THRESHOLD = float(ap.GetParameterAsText(4))

# Temporary output feature class variables
TEMP_TOURIST_SINGLE = "TEMP_TOURIST_SINGLE"
```

```
TEMP_TOURIST_LINE = "TEMP_TOURIST_LINE"
TEMP_ROADS_POINTS = "TEMP_ROADS_POINTS"
TEMP_OUTPUT_POINT = "TEMP_OUTPUT_POINT"

# List of Temporary layers to be deleted at the end of the script
layers_to_delete = [
    "TEMP_TOURIST_SINGLE",
    "TEMP_TOURIST_LINE",
    "TEMP_ROADS_POINTS",
    "TEMP_OUTPUT_POINT"
]
```

5.1.2 Definice funkce

V druhé sekci kódu je definována funkce, která přichytává (snapuje) vstupní body na cílené body z druhé vstupní vrstvy. Funkce „**snap_points_to_nearest**“ má tři parametry:

- points,
- target_features,
- max_distance.

První parametr slouží ke vstupu bodů, které si uživatel přeje přichytit na body v druhém parametru „**target_features**“. Třetí parametr určuje maximální vzdálenost bodů, ve které se funkce pokusí najít nejbližší prvek ve vrstvě „**target_features**“ pro každý bod z vrstvy „**points**“.

Funkce pracuje tak, že nejdříve je nastavena proměnná „**search_radius**“ na hodnotu maximální vzdálenosti ve formátu metrů. Následně je použit nástroj „**Near**“ z knihovny **ArcPy** k výpočtu vzdálenosti dvěma vstupními body. Jakmile jsou vzdálenosti určeny, dojde k iteraci pomocí „**UpdateCursor**“ a pro každý bod ze vstupní vrstvy „**points**“ jsou provedeny následující kroky:

- Pokud je vzdálenost mezi body menší nebo rovna „**max_distance**“, bod se "přichytí" ke nejbližšímu prvku ve vrstvě „**target_features**“.
- Pokud je vzdálenost větší než „**max_distance**“, bod zůstává na svém místě a jeho poloha se nemění.

S koncem funkce jsou aktualizovány polohy bodů na základě provedených přichycení. V rámci vývoje této funkce byla část inspirována Python kódem od dr. Robina Wilsona, který dal základy nástroji pro přichycení bodů na nejbližší liniové prvky (Wilson, 2010).

```
# Definition of function
def snap_points_to_nearest(points, target_features, max_distance):
    ''' This function calculates the distance between points and target_features in
order to snap them. '''
    search_radius = "{} Meters".format(max_distance)
    ap.analysis.Near(points, target_features, "", "LOCATION", "NO_ANGLE", "GEODESIC")

    fields = ["NEAR_X", "NEAR_Y", "NEAR_DIST", "SHAPE@"]
    with ap.da.UpdateCursor(points, fields) as cursor:
        for row in cursor:
            near_x, near_y, near_dist, shape = row

            # If the distance is less than max_distance, perform snap
```

```
            if near_dist <= max_distance:
                new_x, new_y = near_x, near_y
            else:
                # If the distance is greater than max_distance, preserve the original
position of the point
                new_x, new_y = shape.firstPoint.X, shape.firstPoint.Y

            # Create a new point with the snapped values
            new_point = ap.Point(new_x, new_y)

            # Update the shape of the point
            row[3] = new_point

            # Update the row data
            cursor.updateRow(row)
```

5.1.3 Funkční část skriptu

Jakmile je definována funkce, je možné přistoupit k poslední části kódu. Díky možnosti nastavení filtru v **ArcGIS Pro** je zakomentována část, která kontroluje, zda jsou vstupní vrstvy liniové. Jako první jsou tedy kontrolovány dvě podmínky, které zajišťují správný chod skriptu. Následně se provádí segmentace vstupní vrstvy z **multipart** na **singlepart**, což znamená, že pokud jsou vstupní prvky spojené (**multipart**), budou rozděleny na jednotlivé prvky (**singlepart**). Podél obou vstupních vrstev jsou v další části generovány body ve vzdálenosti tří metrů od sebe. Tato hodnota mezery byla vybrána na základě testování různých druhů datových zdrojů a výsledků skriptů. Hodnoty mohou být uživatelem samozřejmě upraveny. Obecně platí, že čím větší mezery mezi generovanými body, tím je menší časová náročnost skriptu. Opačně je to však v případě uživatelem zadané limitní hodnoty pro přichycení bodů. S větším „**thresholdem**“ přibývá potenciálních bodů, které by mohly být snapnuty. Po vygenerování bodů podél linií jsou body s pomocí funkce „**snap_points_to_nearest**“ přichyceny a následuje část, kde se funkcionalita dvou vyvíjených kódů v této části práce rozchází. Kód níže je určen pro zpracování dat z OSM, která obsahují informace o značených trasách, tedy hlavně atributy „**route**“, „**colour**“ a „**osmc_symbo**“. Bodová vrstva, která byla přichycována je převedena na vrstvu liniovou a sloučena podle hodnot v poli „**osm_id**“. Zároveň jsou převedeny atributy z originální vstupní vrstvy. Názvy polí jsou také aktualizovány a jako poslední jsou smazány dočasné vrstvy a vyobrazena hláška o úspěšném průběhu skriptu. V případě univerzálního skriptu je kód ukončen po převedení bodů zpět na linie, přenesení atributů a smazání dočasných vrstev.

```
if INPUT_ROADS == INPUT_TOURIST:
    ap.AddError("Input layers must be diffetent. Please use two different
datasets.")
    exit()

if INPUT_ROADS == OUTPUT_TOURIST or INPUT_TOURIST == OUTPUT_TOURIST:
    ap.AddWarning("Output layer has same name as one of input layers. Output
layer will consist of: _output")
    OUTPUT_TOURIST = OUTPUT_TOURIST + "_output"
```

```
# Multipart to singlepart tool for better segmentation
ap.management.MultipartToSinglepart(INPUT_TOURIST, TEMP_TOURIST_SINGLE)
ap.AddMessage("Features have been converted using Multipart to Singlepart tool.")

# Generate points both line features
ap.management.GeneratePointsAlongLines(TEMP_TOURIST_SINGLE, TEMP_OUTPUT_POINT,
'DISTANCE', Distance='3 meters')
ap.management.GeneratePointsAlongLines(INPUT_ROADS, TEMP_ROADS_POINTS,
'DISTANCE', Distance='3 meters')
ap.AddMessage("Points along both input layers have been created.")

# Call snap function and snap points
if not THRESHOLD:
    THRESHOLD = 50
snap_points_to_nearest(TEMP_OUTPUT_POINT, TEMP_ROADS_POINTS, THRESHOLD)
ap.AddMessage("Points have been succesfully snapped.")

# Convert output tourist points back to line features and transfres attributes
ap.management.PointsToLine(TEMP_OUTPUT_POINT, TEMP_TOURIST_LINE, "ORIG_FID")
ap.JoinField_management(TEMP_TOURIST_LINE, "ORIG_FID", TEMP_OUTPUT_POINT,
"ORIG_FID")
ap.AddMessage("Points features have been converted back to line features and
attribute fields have been transfered.")

if OSM_DATA == "true":
# Dissolve by field to reduce the number of features
ap.analysis.PairwiseDissolve(TEMP_TOURIST_LINE, OUTPUT_TOURIST, ["osm_id"],
[["route", "FIRST"], ["colour", "FIRST"], ["osmc_symbo", "FIRST"]], "MULTI_PART")

# Change field names
for old_field, new_field in zip(["FIRST_route", "FIRST_colour",
"FIRST_osmc_symbo"], ["route", "colour", "osmc_symbo"]):
ap.management.AlterField(OUTPUT_TOURIST, old_field, new_field, clear_field_alias
= "CLEAR_ALIAS")
ap.AddMessage("Line features have been dissolved by osm_id field and names of
attribute fields have been properly altered.")
else:
    ap.management.CopyFeatures(TEMP_TOURIST_LINE, OUTPUT_TOURIST)

# Delete temporary layers
for layer_name in layers_to_delete:
    ap.management.Delete(layer_name, "FeatureClass")
ap.AddMessage("Temporary layers have been deleted.")

# End of the script
ap.AddMessage("Script has run successfully.")
```

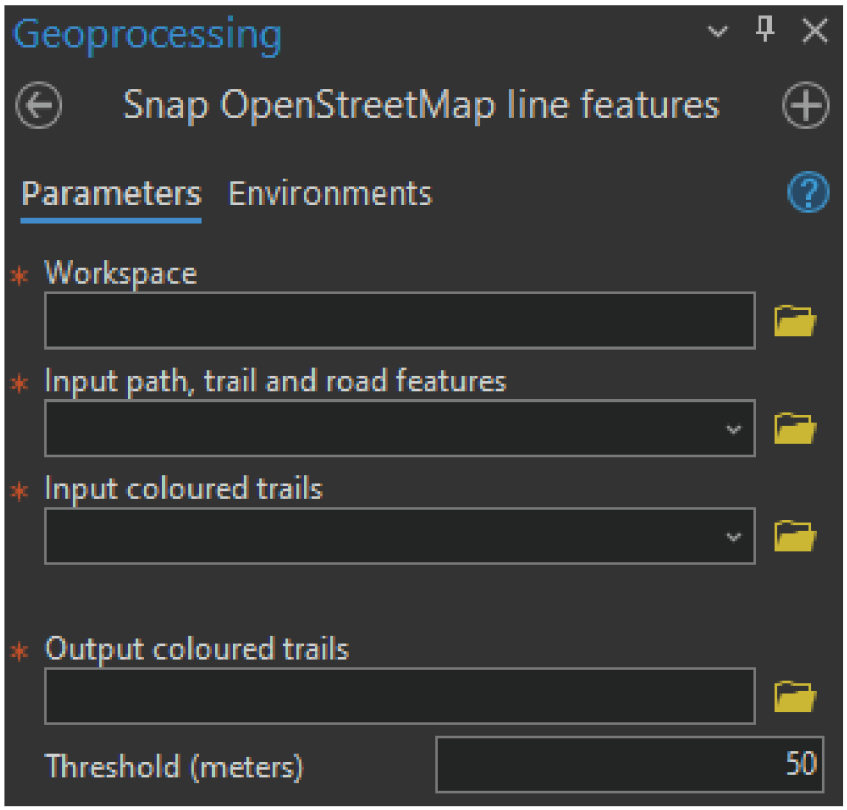
5.2 Převod do ArcGIS Pro

Během vývoje skriptu v prostředí programu **Visual Studio Code** byl kód testován s pomocí vytvořených nástrojů v programu **ArcGIS Pro**. Nástrojům byly opět nastaveny parametry, viz tabulka 3. Se správným nastavením parametrů bylo možné začít skript testovat. Parametry pro univerzální funkci jsou pojmenovány rozdílně oproti nástroji pro OSM data.

Tab. 3: Nastavení vstupních parametrů v ArcGIS Pro

Name	**Data Type**	**Type**	**Direction**	**Filter**	**Default**
Workspace	Workspace	Required	Input		
Input_path_trail_and_road_features	Feature layer	Required	Input	Feature type	
Input_coloured_trails	Feature layer	Required	Input	Feature type	
Output_coloured_trails	Feature Class	Required	Output		
Threshold	Long	Optional	Input	Range	50

Z tabulky lze vyčíst, že jsou vstupním parametrům nastaveny určité filtry. V kapitole 4 jsou tyto filtry popsány podrobněji. Pro dvě vstupní vrstvy je použito nastavení, že uživatel může nahrát pouze liniové vrstvy. A pro „**Threshold**“ je použit datový typ „**Long**“ který uživateli dovolí zadat pouze celá čísla s tím, že minimální hodnota je 1. Výchozí hodnota pro tento parametr byla nastavena na 50 metrů. Na obrázku níže (obr. 19) lze vidět všechny vstupní parametry nástroje, které byly nastavovány.

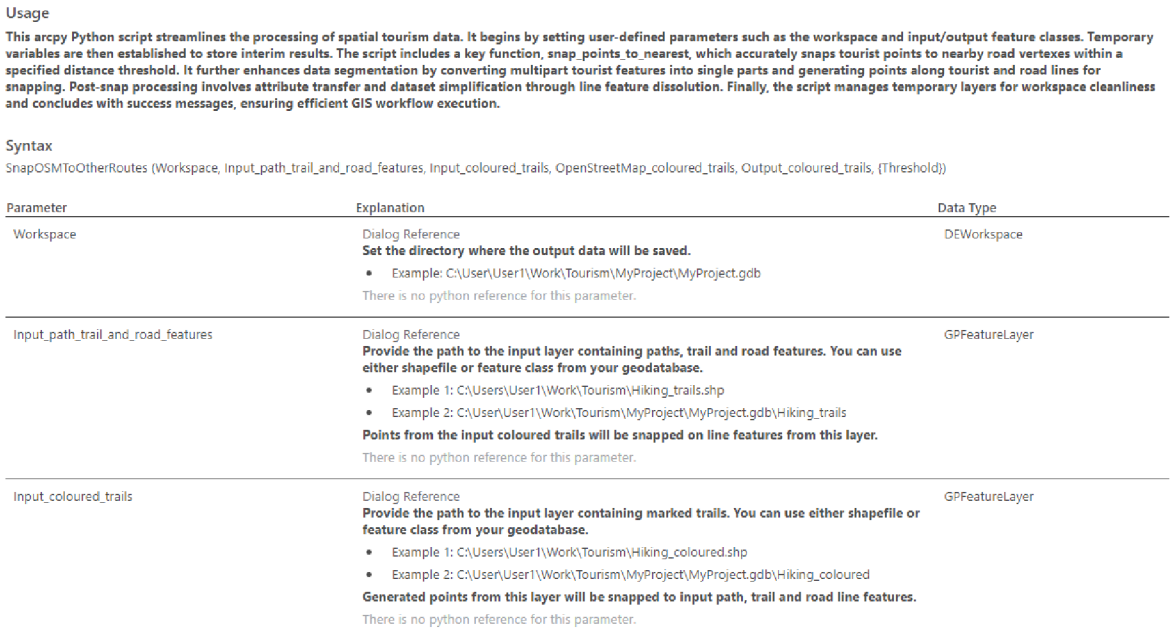


Obrázek 19 – Nástroj na přichytávání linií z OSM se vstupními parametry

5.3 Dokumentace

Dokumentace byla opět prováděna po vzoru kapitoly 4 a její podkapitoly 4.4. Do skriptu byly přidávány zakomentované řádky informující uživatele o jednotlivých krocích. Zároveň byla vytvořena dokumentace na začátku skriptu obsahující základní informace, pseudokód, popis využití a nastavení parametrů v prostředí **ArcGIS Pro**.

Nejdůležitější částí dokumentace byla provedena právě v **ArcGIS Pro**. Zde byly přidány nápovědy pro uživatele a vytvořena metadata s ilustračními obrázky. Na obrázku níže (obr. 20) lze vidět část sekce metadat.



Usage

This arcpy Python script streamlines the processing of spatial tourism data. It begins by setting user-defined parameters such as the workspace and input/output feature classes. Temporary variables are then established to store interim results. The script includes a key function, snap_points_to_nearest, which accurately snaps tourist points to nearby road vertexes within a specified distance threshold. It further enhances data segmentation by converting multipart tourist features into single parts and generating points along tourist and road lines for snapping. Post-snap processing involves attribute transfer and dataset simplification through line feature dissolution. Finally, the script manages temporary layers for workspace cleanliness and concludes with success messages, ensuring efficient GIS workflow execution.

Syntax

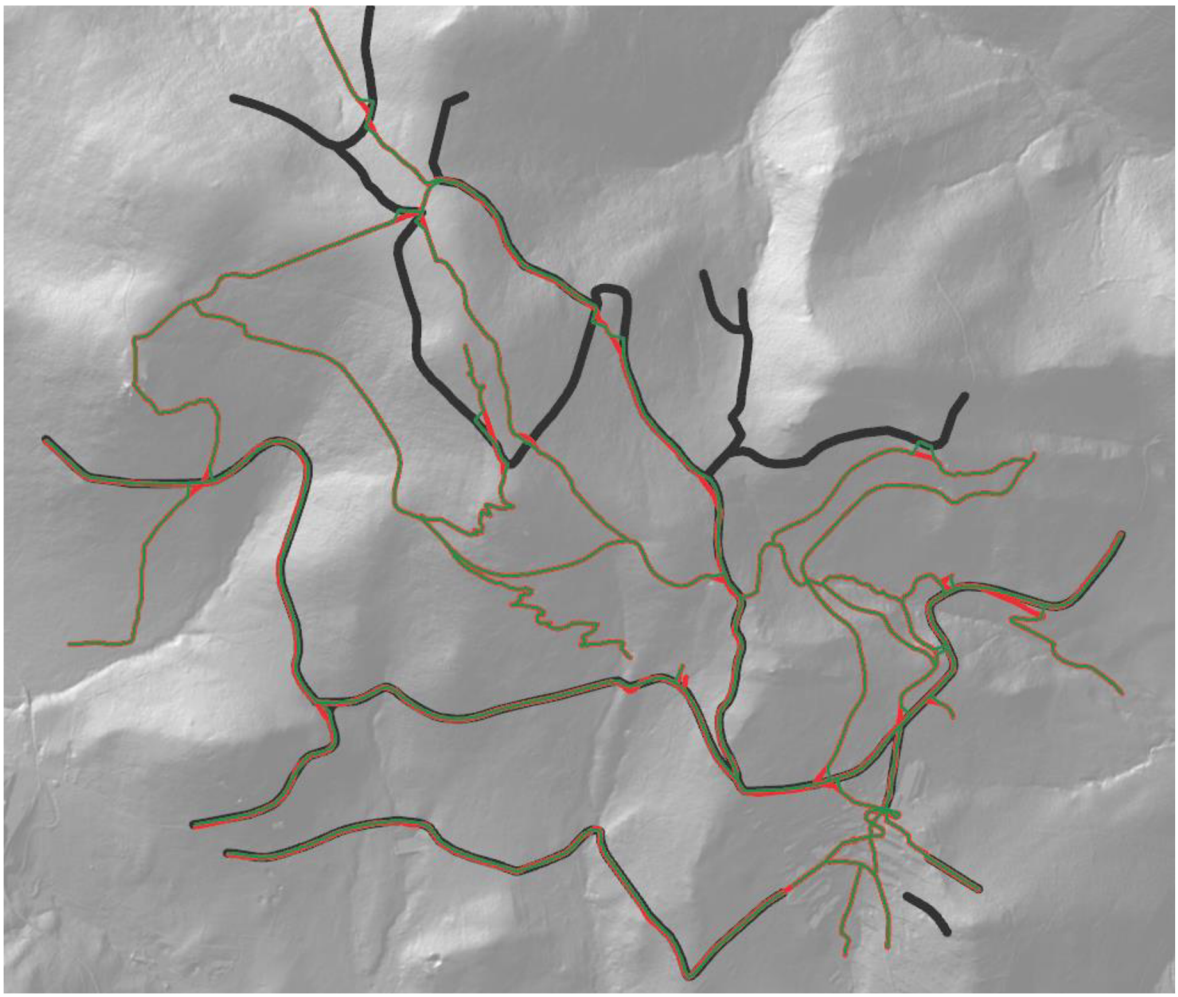
SnapOSMToOtherRoutes (Workspace, Input_path_trail_and_road_features, Input_coloured_trails, OpenStreetMap_coloured_trails, Output_coloured_trails, {Threshold})

Parameter	Explanation	Data Type
Workspace	Dialog Reference **Set the directory where the output data will be saved.** • Example: C:\User\User1\Work\Tourism\MyProject\MyProject.gdb There is no python reference for this parameter.	DEWorkspace
Input_path_trail_and_road_features	Dialog Reference **Provide the path to the input layer containing paths, trail and road features. You can use either shapefile or feature class from your geodatabase.** • Example 1: C:\Users\User1\Work\Tourism\Hiking_trails.shp • Example 2: C:\User\User1\Work\Tourism\MyProject\MyProject.gdb\Hiking_trails **Points from the input coloured trails will be snapped on line features from this layer.** There is no python reference for this parameter.	GPFeatureLayer
Input_coloured_trails	Dialog Reference **Provide the path to the input layer containing marked trails. You can use either shapefile or feature class from your geodatabase.** • Example 1: C:\Users\User1\Work\Tourism\Hiking_coloured.shp • Example 2: C:\User\User1\Work\Tourism\MyProject\MyProject.gdb\Hiking_coloured **Generated points from this layer will be snapped to input path, trail and road line features.** There is no python reference for this parameter.	GPFeatureLayer

Obrázek 20 – Část metadat nástrojů

5.4 Testování

Vyvíjené skripty byly opět testovány na zkušebních datech. Testování začalo na autorem vytvořených vrstvách se základními a jednoduchými liniemi. Následovala objemnější data, která rovněž trvala delší dobu na zpracování. V případě, že bylo testováno snapování linií z OSM na vrstvu cest z databáze Data50, byl výsledný čas nástroje okolo patnácti minut pro oblast okolí města Jeseník do vzdálenosti 20 km. Obrázek níže (obr. 21) demonstruje testování skriptu v oblasti přibližně 4 × 4 km. Černou barvou jsou zobrazovány linie Data50, na které byly přichycovány červené linie reprezentující OSM data. Výsledná zelená barva linií reprezentuje výsledek nástroje.



Obrázek 21– Testování nástroje pro přichycování linií

Při pohledu na větší detail (obr. 22) lze sponzorovat rozdíly jednotlivých vrstev. Na prvním obrázku je patrný rozdílný průběh dvou vstupních vrstev. Druhý obrázek je tvořen výslednou vrstvou (zelená barva), která byla přichycena na požadovanou liniovou vrstvu (Data50). Z posledního obrázku je patrný posun původních červených linií (OSM), které tvořily vstupní vrstvu, a do podoby zelených výstupních linií .

Obrázek 22 – Porovnání liniových vrstev

6 VÝSLEDKY

Hlavními výsledky této diplomové práce jsou skripty a nástroje na automatizaci zobrazení značených tras na mapách. V rámci příloh jsou tyto skripty k dispozici. V hlavní, tedy elektronické podobě jsou skripty a nástroje (doplněné o ukázkový project package) ke stažení na webu Katedry geoinformatiky UP v sekci „**Pro studenty**“ a následně „**Diplomové práce**“.

V rámci vyvíjení kódů byly následovány dílčí kroky popsané v kapitole 3. Nejdříve bylo nutné provést rešerši dané problematiky a postupně začít s testováním Python knihoven a jejich funkcionalit. Po provedení rešerše a testování knihoven bylo rozhodnuto, že jediná knihovna, která bude v rámci skriptů a nástrojů používána je **ArcPy**.

S postupným vývojem byly skripty testovány v prostředí **AcrGIS Pro** v rámci vytvořených nástrojů. Tyto nástroje umožňují importovat Python kód a po nastavení vstupních parametrů a filtrů fungují tak, jako klasické nástroje. Testování probíhalo na velkém množství dat, primárně ale na datech z databáze **OSM**, jejichž součástí jsou právě informace o barevně značených turistických trasách. Zároveň byly tvořeny i vlastní datové zdroje a nástroje byly testovány i na nich. Díky uživatelsky přívětivé možnosti stahovat data z **OSM** s pomocí **QGIS** a pluginu **QuickOSM** je možné zautomatizovat zobrazování značených tras na mapách.

Hlavním výsledkem této práce jsou dva skripty a jedna sada nástrojů určené pro **ArcGIS Pro**. V případě, že by uživatel nepoužíval prostředí **ArcGIS Pro**, má možnost použití pouze skriptů, ve kterých jsou popsány veškeré důležité informace o použití. Uživatelsky přívětivější je ale využití v rámci nástroje v **ArcGIS Pro**. Uživateli jsou zde zobrazovány nápovědy a zároveň jsou díky filtrům ošetřeny vstupní parametry tak, aby byla co největší šance úspěšného zakončení nástroje. I díky hláškám generovaným s pomocí funkcí knihovny **ArcPy** je uživatel informován o postupném průběhu skriptu. V případě chyby tedy uživatel rovnou ví, ve které části nástroje nastala chyba.

Výslednými hlavními skripty a nástroji tedy jsou:

- skript pro generování turistických tras společně s cyklostezkami,
- skript pro generování turistických tras společně s lyžařskými trasami,
- nástroj pro generování turistických tras společně s cyklostezkami,
- nástroj pro generování turistických tras společně s běžeckými trasami.

S pomocí těchto skriptů a nástrojů je možné v mapách v prostředí **ArcGIS Pro** generovat barevné turistické trasy společně s cyklistickými nebo běžeckými trasami. Právě z důvodu generování cyklistických a běžeckých tras byla zvolena tvorba tzv. **letní a zimní varianty** nástrojů.

V zadání práce je dále zmíněna i varianta, kdy je potřeba upravit průběh značených tras podle průběhu linií z druhé datové sady. I tato varianta byla nakonec zvážena. Byly vytvořeny **skripty a nástroje**, které po definování vstupních parametrů upravují průběh linií se značenými trasami a přichycují (**snapují**) jednotlivé body na body definované druhou vstupní vrstvou s pomocí vyvinuté funkce. Výsledku bylo docíleno tak, že se podél obou linií generují body, které jsou následně na sebe přichycovány, a nakonec je opět výsledkem linie, obsahující informace o značených trasách. Druhý skript je zároveň univerzální a není tedy nutné jej používat jen na data z **OSM**. Výsledné skripty byly později rovněž převedeny na nástroje v **ArcGIS Pro** a byly jim přidány nápovědy a hlášky informující o průběhu kódu.

Výslednými skripty a nástroji této sekce jsou:

- skript pro přichycení liniových dat z OSM na liniová data z druhé datové sady,
- univerzální skript pro přichycení liniových na liniová data z druhé datové sady,
- nástroj pro přichycení liniových dat z OSM na liniová data z druhé datové sady,
- univerzální nástroj pro přichycení liniových na liniová data z druhé datové sady.

Díky vývoji výše zmíněných nástrojů je možné je propojit s hlavními nástroji vytvořenými v rámci této diplomové práce. V případě, že je potřeba nejdříve vstupní data značených tras přichytit na jiný datový zdroj, lze spustit nástroj, který předpřipraví data pro nástroj na generování značených tras. Vznikly tak nástroje, které nemusí být nutně spjaty jedny s druhými, ale je možné je využít i samostatně, a pro velké množství dat. I přes to, že jsou nástroje primárně cíleny na data z **OSM**, je možné je využít i pro ostatní datové zdroje v případě splnění podmínek důležitých pro běh nástrojů.

Výsledkem této práce jsou tedy čtyři skripty a dvě sady nástrojů pro ArcGIS Pro, z nichž každá sada obsahuje dva nástroje. Zároveň byl vytvořen i ukázkový **ArcGIS Pro project package** s testovacími daty a sadami nástrojů. Spolu s výsledky byla vytvořena i ukázková mapa v měřítku 1 : 40 000 zobrazující oblast Hrubého Jeseníku společně se značenými turistickými trasami a cyklostezkami. V této mapě je demonstrována funkcionalita dvou vyvinutých nástrojů. Vstupní liniová data z OSM byla přichycena na vrstvu Data50 a následně byly vygenerovány trasy s pomocí hlavního nástroje na tvorbu barevně značených turistických tras. Mapa byla po finalizaci vytištěna a je k dispozici jako volná příloha společně s možností zobrazení na stránkách Katedry Geoinformatiky Univerzity Palackého.

V rámci povinných příloh k této diplomové práci byl vytvořen a odevzdán i web stručně informující o diplomové práci společně se skripty, nástroji, textem práce a mapou ke stažení. Poslední povinnou přílohou byl pak poster o formátu A2.

7 DISKUZE

Tvorba výsledných skriptů a nástrojů byla ve srovnání s původním očekáváním autora náročnější hlavně z důvodu omezených znalostí programovacího Jazyka Python a syntaxe, která byla ovšem řádně nastudována. Původně jednoduchá myšlenka se vyvinula v komplexní problematiku. Během procesu tvorby bylo nutné mnohokrát restartovat a pouštět nástroje opětovně. Dalo by se říct, že testování kódů proběhlo ve stovkách pokusů.

Debugging

Programování v rozsahu práce bylo pro autora velkou výzvou. Mnohokrát zobrazované chybové hlášky a jejich řešení byly v některých případech velice časově náročné. Naopak díky skvělému propojení **ArcGIS Pro** a **Pythonu** byla práce příjemně zlehčována. Pro ostatní GIS produkty by vývoj nástrojů mohl být složitější. Právě díky chybovým hláškám byl autor schopen naučit se nové dovednosti ve vývoji nástrojů do **ArcGIS Pro**.

Téma

Téma jako takové bylo autorovi velice blízké. Autorem milovaná turistika a mapy byly výbornou motivací pro zpracování této diplomové práce. Problematiku vykreslování tras autor řešil již u bakalářské práce a časová náročnost byla tehdy obrovská. Díky vývoji nástrojů pro generování barevně značených tras ušetří uživatelé čas s tvorbou nejen turistických map, ale jakýchkoliv map, kde je potřeba generování paralelních linií.

Tvorba ukázkové mapy

Tvorba ukázkové mapy byla jakousi odměnou po odvedené práci na vývoji skriptů. S pomocí různých datových zdrojů byla vytvořena ukázková mapa, která demonstruje funkcionalitu vyvinutých nástrojů. Autorem byly také pořízeny dvě turistické mapy od českých vydavatelů, které sloužily jako inspirace pro tvorbu výsledné mapy. Ukázková mapa má samozřejmě prostor pro zlepšení. Bylo by například potřeba lepších datových zdrojů. Hlavní myšlenkou bylo ale demonstrování automatizace zobrazování turistických tras na mapách.

Tvorba webu a posteru

Tvorba webu a posteru probíhala jako poslední jakožto povinná část této diplomové práce. Jak web, tak i poster zkráceně informují o zadání, cíli, průběhu a výsledcích. Poster byl tvořen s nahlédnutím na grafické trendy dnešní doby.

Limitace

Práce má samozřejmě své limitace, které by mohly být do budoucna řešeny. Například nástroj Repair Self Intersection, který není licencován v základní verzi ArcGIS Pro. Výhledem do budoucna by bylo vytvoření nástroje, který by linie, které protínají samy sebe opravoval. Nástroj pro přichycování linií by rovněž mohl být optimalizován více tak, aby proběhl rychleji. Účel této práce ale nástroj splňuje, a navíc ještě předchází.

Potenciál této práce a výhled do budoucna

Autor věří, že tato práce má do budoucna velký potenciál. Všechny vytvořené nástroje lze vyvíjet a upravovat dále podle potřeb uživatelů a díky nápovědám a zpracované dokumentaci by to pro uživatele nemuselo představovat větší problém. Automatizace vykreslování značených tras je při tvorbě turistických map velice důležité, jelikož tvorba těchto druhů map je časově velice náročná. Nástroj na přichycování linií z různých datových zdrojů má do budoucna možná ještě větší potenciál přesahující tvorbu

turistických tras. Pokud by byl tento nástroj dále vyvíjen, jeho využití by mohlo být přínosné pro řadu geoinformatických operací.

Celkové zhodnocení

Tvorbu této diplomové práce provázela řada komplikací, kterým bylo buďto nutné se přizpůsobit, nebo je vyřešit. S přibývající funkcionalitou a testováním přibývalo i chybových hlášek, které musely být opravovány. Nakonec mají ale skripty a nástroje velký potenciál využitelnosti do budoucna. Vyvinuté nástroje uživatelům zjednoduší a zpříjemní tvorbu turistických map a také řešení problémů s liniovými vrstvami, které nemají totožný průběh. Na skriptech a nástrojích je samozřejmě možné dále pracovat a vyvíjet je podle potřeb koncových uživatelů.

8 ZÁVĚR

Hlavní myšlenkou a cílem zpracování této diplomové práce bylo zlepšení procesu tvorby turistických map. Díky tvorbě nástrojů popsaných v této práci je možné identifikovat barvy značených tras a automaticky je vložit do mapy za sebe v daném pořadí. Touto prací byla vyřešena problematika generování barevně značených tras paralelně s pomocí kombinatoriky a proces tvorby těchto tras byl signifikantně zrychlen. Díky databázi **OSM**, jejíž součástí jsou i data o značených trasách bylo možné dosáhnout stanovených cílů pro tuto práci a vyvinout výsledné nástroje.

Práce byla rozdělena na teoretickou i praktickou část, kde teoretická tvořila hlavně studování dat, Pythonu a knihoven pro následnou praktickou část. V té byly vyvíjeny skripty, které byly později vloženy do prostředí **ArcGIS Pro**, kde z nich byly vytvořeny nástroje pro automatizaci výše zmíněných procesů.

Díky vyvinutým nástrojům je tedy možné nejen zobrazovat správně barevně značené trasy v mapách, ale také kombinovat liniové vrstvy z různých datových zdrojů, jejichž průběhy jsou v mnoha případech rozdílné.

Tato práce poskytuje vědecký přístup k řešení problémů v rámci vizualizace značených tras v mapách, a to jak v desktopových, tak tištěných. Právě díky programátorských a geoinformatických komunit bylo možné tuto práci zpracovat. Podrobně zpracované dokumentace a ukázky stály rovněž za úspěchem dosažení kýženého výsledku při tvorbě skriptů a nástrojů.

Limitace a prostor ke zlepšení jsou rovněž součástí této práce. Pro lepší fungování nástrojů by bylo potřeba vyvinout nástroj, který dokáže měnit, odkud a kam byla linie tvořena. V určitých případech je možné že se turistické a cyklistické trasy mohou generovat přes sebe, a to hlavně v případě rozdílných datových zdrojů. Linie jsou totiž generovány na základě informace, zda se mají generovat na levé, nebo pravé straně linie. Další limitací je nástroj „**RepairSelfIntersection**“, který není součástí základní verze **ArcGIS Pro**, a proto musí být licencován za poplatek. Jako další vylepšení by tedy bylo možné vyvinout nástroj, který opraví linie, jenž protínají samy sebe.

Závěrem je nutné je vyzdvihnout hlavně všestrannost skriptů a nástrojů. Díky dvěma možnostem vizualizace je možné uspokojit požadavky uživatele. Původně měly tyto nástroje být jen pro data z **OSM**, ale díky dalšímu vývoji bylo možné přijít i s řešením pro datové sady, které jsou svou strukturou dat a atributů jiné než data **OSM**.

Výsledkům bylo dosaženo také díky skvělé funkcionalitě knihovny **ArcPy** a dobré provázanosti s **ArcGIS Pro**. Vytvořené nástroje jsou pro výsledného uživatele mnohem přívětivější, než používání skriptů nebo Python okna.

Nejdůležitější myšlenkou celé práce ale bylo přispění geoinformatické komunitě s novými nástroji pro zpracování a zobrazování značených tras v mapách. Výsledné nástroje a skripty by mohly posloužit jako základní stavební kameny dalších nejen akademických prací. Potenciál vyvinutého nástroje pro přichycení linií z různých datových zdrojů je možná ještě větší než potenciál nástroje pro tvorbu barevně značených tras. S rozvojem turismu a značených tras má ale i hlavní nástroj velký potenciál do budoucna.

POUŽITÁ LITERATURA A INFORMAČNÍ ZDROJE

BRABEC TOMÁŠ, 2014. Klub českých turistů v letech 1888 - 1918 [online]. B.m. [vid. 2024-05-08]. b.n. Dostupné z: https://dspace.cuni.cz/handle/20.500.11956/70797?show=full

CAMBRIDGE UNIVERSITY, 2024. tourism [online] [vid. 2024-05-08]. Dostupné z: https://dictionary.cambridge.org/dictionary/english/tourism#google_vignette

CK MUNDO, 2024. Hory a outdoor ve Španělsku [online] [vid. 2024-05-08]. Dostupné z: https://www.mundo.cz/spanelsko/hory

ČT24, mka, 2016. České turistické trasy jsou světový unikát, chránit by je mohl samostatný zákon [online] [vid. 2024-05-08]. Dostupné z: https://ct24.ceskatelevize.cz/clanek/domaci/ceske-turisticke-trasy-jsou-svetovy-unikat-chranit-by-je-mohl-samostatny-zakon-104888

ESRI, 2020. Create parallel features similar to the Copy Parallel tool in ArcGIS Pro with Python [online] [vid. 2024-05-08]. Dostupné z: https://support.esri.com/en-us/knowledge-base/how-to-create-parallel-features-similar-to-the-copy-par-000024330

ESRI, 2024a. Python in ArcGIS Pro [online] [vid. 2024-05-08]. Dostupné z: https://pro.arcgis.com/en/pro-app/latest/arcpy/get-started/installing-python-for-arcgis-pro.htm

ESRI, 2024b. What is ArcPy? [online] [vid. 2024-05-08]. Dostupné z: https://pro.arcgis.com/en/pro-app/latest/arcpy/get-started/what-is-arcpy-.htm

EUROPEAN COMMISION, 2019. New guidance for European cycling projects now available [online] [vid. 2024-05-08]. Dostupné z: https://urban-mobility-observatory.transport.ec.europa.eu/news-events/news/new-guidance-european-cycling-projects-now-available-2019-07-01_en?prefLang=sv

EUROPEAN RAMBLERS ASSOCIATION, 2022. Waymarking in Europe [online]. 4th Edition. B.m.: European Union of Mountaineering Associations (EUMA) & European Ramblers' Association (ERA) [vid. 2024-05-08]. Dostupné z: https://www.european-mountaineers.eu/storage/app/media/Project%20and%20public-documents/Erasmus%20plus%20good%20governance/2022Waymarking_in_Europe_4th_Edition_E-Book-final-3.pdf

FLOGNFELDT JR., Thor, 2005. The tourist route system – models of travelling patterns. Belgeo [online]. (1–2), 35–58. ISSN 1377-2368. Dostupné z: doi:10.4000/belgeo.12406

FREEMAP SLOVAKIA, 2022. freemap.sk [online] [vid. 2024-05-08]. Dostupné z: https://www.freemap.sk/#map=13/49.101803/20.058632&layers=X

GALVASOVÁ, Iva and Binek, Jan and Holeček, Jan and Chabičovská, Kateřina and Szczyrba, Zdeněk and kol, a, 2008. Průmysl cestovního ruchu [Tourism Industry] [online]. [vid. 2024-05-08]. Dostupné z: https://www.researchgate.net/publication/273136559_Prumysl_cestovniho_ruchu_Tourism_Industry

GLONEK, Jiří, 2019. Na Jeseníky! : o putování jesenickými horami, turistických bedekrech a mapách do roku 1945.

GUO ZHAOWEN, 2024. About the Silk Roads [online] [vid. 2024-05-08]. Dostupné z: https://en.unesco.org/silkroad/about-silk-roads

IONOS, 2023. Python vs. Java: What's the difference? [online] [vid. 2024-05-08]. Dostupné z: https://www.ionos.com/digitalguide/websites/web-development/python-vs-java/

JANCEWICZ, Kacper a Dorota BOROWICZ, 2017. Tourist maps – definition, types and contents. Polish Cartographical Review [online]. 49(1), 27–41. ISSN 2450-6966. Dostupné z: doi:10.1515/pcr-2017-0003

KADLČÁKOVÁ E., Poláková A., 2016. Turistické značky máme od roku 1889, jsou geniálně jednoduché a závidí nám je celý svět [online] [vid. 2024-05-08]. Dostupné z: https://budejovice.rozhlas.cz/turisticke-znacky-mame-od-roku-1889-jsou-genialne-jednoduche-a-zavidi-nam-je-7046780

KARNICK, P., D. CLINE, S. JESCHKE, A. RAZDAN a P. WONKA, 2010. Route Visualization Using Detail Lenses. IEEE Transactions on Visualization and Computer Graphics [online]. 16(2), 235–247. ISSN 1077-2626. Dostupné z: doi:10.1109/TVCG.2009.65

KLAUZ MICHAL, 2020. Vývoj turistického značení na bývalém území prvorepublikového Československa [online]. Praha [vid. 2024-05-08]. UNIVERZITA KARLOVA V PRAZE. Dostupné z: https://dspace.cuni.cz/handle/20.500.11956/38215

KLUB ČESKÝCH TURISTŮ, 1916. Časopis turistů.

KLUB ČESKÝCH TURISTŮ, 2012. VÝVOJ TURISTICKÉHO ZNAČENÍ u nás a značení turistických tras ve většině evropských zemí [online] [vid. 2024-05-08]. Dostupné z: https://www.kct-jmo.cz/sites/default/files/users/user1/dokumenty/znaceni/vyvoj-turistickeho-znaceni.pdf

KLUB ČESKÝCH TURISTŮ, 2013. ZÁKLADNÍ PRAVIDLA ZNAČENÍ TURISTICKÝCH TRAS [online] [vid. 2024-05-08]. Dostupné z: https://kct-rokytnice.com/wp-content/uploads/2019/06/Mal%C3%A1_p%C5%99%C3%ADru%C4%8Dka_vyti%C5%A1t%C4%9Bn%C3%A1-verze_120x170.pdf

KLUB ČESKÝCH TURISTŮ, 2023. Mongolsko začalo používat české značení turistických tras [online] [vid. 2024-05-08]. Dostupné z: https://kct.cz/clanky/mongolsko-zacalo-pouzivat-ceske-znaceni-turistickych-tras

KLUB ČESKÝCH TURISTŮ, 2024a. Czech Tourist Club (KČT) [online] [vid. 2024-05-08]. Dostupné z: https://kct.cz/english

KLUB ČESKÝCH TURISTŮ, 2024b. Historie KČT [online] [vid. 2024-05-08]. Dostupné z: https://kct.cz/historie

KLUB ČESKÝCH TURISTŮ, 2024c. Systém turistického značení [online] [vid. 2024-05-08]. Dostupné z: https://kct.cz/system-turistickeho-znaceni

KOSTADINOV, Georgi, Kristian MILEV, Stefan STAYNOV a Asya STOYANOVA-DOYCHEVA, 2020. Algorithm for Genrerating and Visualizing Routes of an Intelligent Tourist Guide. In: 2020 International Conference Automatics and Informatics (ICAI) [online]. B.m.: IEEE, s. 1–5. ISBN 978-1-7281-9308-3. Dostupné z: doi:10.1109/ICAI50593.2020.9311310

LECHNER RUDOLF, 1892. Spezial-Karte der Mährisch-Schlesischen Sudeten [online]. 1892. [vid. 2024-05-08]. Dostupné z: https://rcin.org.pl/igipz/dlibra/publication/13756/edition/9168/content

MAFRA, 2020. Česko, jak ho neznáte [online]. [vid. 2024-05-08]. Dostupné z: https://www.etema.cz/objednavka-print.aspx?t=lp&id=predplatne-tema/ceskojakhoneznate.htm

MAUDET, Adrien, Guillaume TOUYA, Cécile DUCHÊNE a Sébastien PICAULT, 2017. DIOGEN, a multi-level oriented model for cartographic generalization. International Journal of Cartography [online]. 3(1), 121–133. ISSN 2372-9333. Dostupné z: doi:10.1080/23729333.2017.1300997

MOKROŠOVÁ, Petra, 2010. Vývoj turistických tras a značení na území České republiky [online]. B.m. [vid. 2024-05-08]. b.n. Dostupné z: https://theses.cz/id/r2mm85/

MORGAN KATE, 2023. Forget Google Maps: Why Paper Map Sales Are Booming [online] [vid. 2024-05-08]. Dostupné z: https://www.wsj.com/articles/why-paper-map-sales-are-booming-11674164824

MURRAY, Michael a Brian GRAHAM, 1997. Exploring the dialectics of route-based tourism: the Camino de Santiago. Tourism Management [online]. 18(8), 513–524. ISSN 02615177. Dostupné z: doi:10.1016/S0261-5177(97)00075-7

NEČAS RICHARD, 2018. Značené turistické trasy [online] [vid. 2024-05-08]. Dostupné z: https://turistika-jarcova4.webnode.cz/news/znacene-turisticke-trasy/

NÍDR TOMÁŠ, 2016. České turistické značky najdete i v Brazílii. Možná budou v celé zemi [online] [vid. 2024-05-08]. Dostupné z: https://www.idnes.cz/cestovani/kolem-sveta/ceske-turisticke-znacky-brazilie.A161021_2281003_kolem-sveta_hig

NÖLLENBURG MARTIN, 2014. (PDF) A survey on automated metro map layout methods [online] [vid. 2024-05-08]. Dostupné z: https://www.researchgate.net/publication/281587114_A_survey_on_automated_metro_map_layout_methods

ONYSHCHENKO, Mariia, Vitalii OSTROUKH, Viktoriia LEPETIUK a Iryna PIDLISETSKA, 2022. Creation of Tourist Maps Series as a Type of Regional System Tourism Mapping. The Cartographic Journal [online]. 59(1), 69–82. ISSN 0008-7041. Dostupné z: doi:10.1080/00087041.2021.1937827

PROTASIEWICZ JAKUB, 2024. Comparing Python vs. C Sharp: A Comprehensive Guide for 2024 [online] [vid. 2024-05-08]. Dostupné z: https://www.netguru.com/blog/python-vs-c-sharp

PTÁČEK JAN, 2020. S MAPOU NA CESTÁCH. In: [online]. [vid. 2024-05-08]. Dostupné z: https://kartografickyden.upol.cz/old/prezentace/14_rocnik/ptacek_14kdo.pdf

PUBLISHERSGLOBAL, 2024. Map Publishers [online] [vid. 2024-05-08]. Dostupné z: https://www.publishersglobal.com/directory/media/map-publishers#google_vignette

PYTHON SOFTWARE FOUNDATION, 1997. Comparing Python to Other Languages [online]. [vid. 2024-05-08]. Dostupné z: https://www.python.org/doc/essays/comparisons/

PYTHON SOFTWARE FOUNDATION, 2024. What is Python? Executive Summary [online] [vid. 2024-05-08]. Dostupné z: https://www.python.org/doc/essays/blurb/

REILLY, Derek, Malcolm RODGERS, Ritchie ARGUE, Mike NUNES a Kori INKPEN, 2006. Marked-up maps: combining paper maps and electronic information resources. Personal and Ubiquitous Computing [online]. 10(4), 215–226. ISSN 1617-4909. Dostupné z: doi:10.1007/s00779-005-0043-6

REY, Belén, Rafael L. MYRO a Asun GALERA, 2011. Effect of low-cost airlines on tourism in Spain. A dynamic panel data model. Journal of Air Transport Management [online]. 17(3), 163–167. ISSN 09696997. Dostupné z: doi:10.1016/j.jairtraman.2010.12.004

SEZGIN, Erkan a Medet YOLAL, 2012. Golden Age of Mass Tourism: Its History and Development. In: Visions for Global Tourism Industry - Creating and Sustaining Competitive Strategies [online]. B.m.: InTech. Dostupné z: doi:10.5772/37283

SEZNAM.CZ, 1998. Mapy.cz [online] [vid. 2024-05-08]. Dostupné z: https://mapy.cz/turisticka

SLOVENIAN TOURIST BOARD, 2024. Vysokohorská turistika [online] [vid. 2024-05-08]. Dostupné z: https://www.slovinsko.travel/vysokohorska-turistika

SLOVENSKÝ CYKLOKLUB ŠARIŠ PREŠOV, 2003. Rozdelenie cyklotrás [online] [vid. 2024-05-08]. Dostupné z: http://cyklotrasy.luho.sk/subory/cyklotrasy_menu2.files/rozdelenie_cyklotras.htm

STACK OVERFLOW, 2018. Developer Survey Results 2018 [online] [vid. 2024-05-08]. Dostupné z: https://insights.stackoverflow.com/survey/2018/#technology

STX NEXT, 2024. Python vs. Other Programming Languages [online] [vid. 2024-05-08]. Dostupné z: https://www.stxnext.com/python-vs-other-programming-languages/

SYNEK LUKÁŠ, 2022. Co to vlastně byla Hedvábná stezka? [online] [vid. 2024-05-08]. Dostupné z: https://www.hedvabnastezka.cz/co-to-vlastne-byla-hedvabna-stezka/

TEULADE-DENANTES, Jules, Adrien MAUDET a Cécile DUCHÊNE, 2015. Routes visualization: Automated placement of multiple route symbols along a physical network infrastructure. Journal of Spatial Information Science [online]. (11). ISSN 1948-660X. Dostupné z: doi:10.5311/JOSIS.2015.11.230

TŘÍSKA JAN, 2021. Význam turistických značek a jejich barev [online] [vid. 2024-05-08]. Dostupné z: https://www.umimeporadit.cz/vyznam-turistickych-znacek/

UN TOURISM, 2024a. Glossary of tourism terms.

UN TOURISM, 2024b. Silk Road [online] [vid. 2024-05-08]. Dostupné z: https://www.unwto.org/silk-road

W3SCHOOLS, 2024. Python Introduction [online] [vid. 2024-05-08]. Dostupné z: https://www.w3schools.com/python/python_intro.asp

WILSON ROBIN, 2010. How to: Snap points to lines in ArcGIS with Python [online] [vid. 2024-05-08]. Dostupné z: https://blog.rtwilson.com/how-to-snap-points-to-lines-in-arcgis-with-python/

PŘÍLOHY

SEZNAM PŘÍLOH

Příloha 1 (volná)	**A2 Poster**
Příloha 2 (elektronická)	**Čtyři skripty**
Příloha 3 (elektronická)	**Čtyři nástroje ve formě dvou toolboxů do ArcGIS Pro**
Příloha 4 (volná)	**Ukázková turistická mapa oblasti Hrubého Jeseníku**
Příloha 5 (elektronická)	**Project package s nástroji a ukázkovými daty**
Příloha 6 (elektronická)	**Web**

Struktura zip souboru s přílohami:

Povinná struktura adresářů:

- **Nastroje** (2 toolboxy pro ArcGIS Pro)
- **Poster** (poster ve formátuech PDF a JPG)
- **PPKX_s_nastroji_a_ukazkami** (ArcGIS Pro project package s ukázkovými daty)
- **Skripty** (čtyři Python skripty)
- **Text_Prace** (text formátu MS Word a PDF)
- **Ukazkova_mapa** (mapa ve formátu PDF)
- **Web** (záloha webu)