Statistické vyhodnocení rozsáhlých dat z odpadového hospodářství

Ing. Veronika Šošolíková

Bakalářská práce
2023

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2022/2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Ing. Veronika Šošolíková**
Osobní číslo:	**A20123**
Studijní program:	**B0613A140020 Softwarové inženýrství**
Forma studia:	**Kombinovaná**
Téma práce:	**Statistické vyhodnocení rozsáhlých dat z odpadového hospodářství**
Téma práce anglicky:	**Statistical Assessment of Big Data in Waste Management**

Zásady pro vypracování

1. Osvojte si znalosti spojené se základními pojmy statistického zpracování dat s přihlédnutím na charakter dat z odpadového hospodářství.
2. Navrhněte zaměření statistického zpracování dat poskytnutých spolupracujícím pracovištěm na Vysokém učení technickém v Brně.
3. Zpracujte data a proveďte jejich zpracování ve vybraném software.
4. Proveďte důkladnou diskuzi dosažených výsledků. Vyslovte omezení a limity zvoleného přístupu a vytyčte směry pro případný další výzkum.

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. ANDĚL, J. Základy matematické statistiky. Vyd. 3. Praha: Matfyzpress, 2011. ISBN 978-80-73-8-162-0.
2. MONTGOMERY, D. C. a RUNGER, G. C. Applied statistics and probability for engineers. Wiley (2018). ISBN 978-1-119--40036-3.
3. SUHOV, Y. a KELBERT, M. Probability and statistics by example: Basic probability and statistics. Cambridge University Press (2005). ISBN 978-0-521-84766-7.
4. HOGG, R. V, TANIS, E. A. a Zimmerman, D. L. Probability and statistics inference, 2015. ISBN 978-0-321-92327-1.
5. YUAN, L., Lu, W. a XUE, F. Estimation of construction waste composition based on bulk density: A big data-probability (BD-P) model. Journal of Environmental Management, 292:112822, 2021. DOI: 10.1016/j.jenvman.2021.112822.

Vedoucí bakalářské práce: **Ing. Dušan Hrabec, Ph.D.**
Ústav matematiky

Datum zadání bakalářské práce: **2. prosince 2022**
Termín odevzdání bakalářské práce: **26. května 2023**

L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 7. prosince 2022

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 25. 5. 2023

Ing. Veronika Šošolíková, v.r.
…….
podpis studenta

ABSTRAKT

Bakalářská práce je zaměřena na vývoj nástroje pro statistické zpracování dat z odpadového hospodářství. Cílem práce je vytvořit obecný výpočtový modul, který se snaží na míru předmětné problematiky data zpracovávat. Práce je rozdělena na část teoretickou a praktickou. Teoretická část se věnuje zejména metodám zaměřených na popisnou statistiku (charakteristiky polohy výběrového souboru, odlehlé pozorování aj.). Tvoří východiska pro tvorbu výpočtových funkcí. V praktické části je naprogramován výpočtový modul včetně grafického uživatelského rozhraní. Nad uživatelem vybranou podmnožinou dat základního souboru jsou realizovány výpočtové funkce, zobrazovány grafy a tvořeny mapy.

Klíčová slova:

Odpadové hospodářství, Popisná statistika, Tkinter, Python, GeoPandas, Analýza dat

ABSTRACT

The bachelor's thesis is focused on the development of a tool for the statistical processing of waste management data. The goal of the work is to create a general computing module that tries to process data to the extent of the issue in question. The work is divided into a theoretical and a practical part. The theoretical part is mainly devoted to methods focused on descriptive statistics (characteristics of the location of the sample, outlier observations, etc.). They form the starting point for the creation of computational functions. In the practical part, a calculation module including a graphical user interface is programmed. Computational functions are implemented, graphs are displayed, and maps are created above the user-selected subset of the basic file data.

Keywords:

Waste management, Descriptive statistics, Tkinter, Python, GeoPandas, Data analysis

Mé velké díky patří externímu konzultantovi z VUT v Brně Ing. Radovanu Šomplákovi Ph.D. za cenné rady a připomínky, které bez ohledu na čas a den promptně poskytoval. Rovněž děkuji svému vedoucímu práce Ing. Dušanu Hrabcovi Ph.D., který mi vymyslel takové téma, které mě skutečně bavilo zpracovávat. A oběma pánům děkuji za trpělivost, že to se mnou nevzdali.

Největší poděkování patří celé mé rodině za podporu při studiu. Zejména mému manželovi Petrovi a dcerám Barunce, Adriance a Danielce, kteří se o sebe dokázali navzájem postarat a dali mi prostor pro vytvoření tohoto díla.

Potřebná data z odpadového hospodářství byla poskytnuta v rámci řešení projektu CEVOOH podporovaný Technologickou agenturou české republiky (SS02030008).

Prohlašuji, že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

Každoročně se v rámci zákonné povinnosti shromažďují údaje o produkci a nakládání s odpady v rámci Hlášení souhrnných údajů z průběžné evidence (Vyhláška 273/2021 Sb.). Data od evidentů se následně geograficky agregují pro tvorbu různých statistik. Nejnižší územní členění je tvořené tzv. základními územními jednotkami (ZÚJ) České republiky. Analýzou a statistickým vyhodnocením těchto dat se dají odhalit skryté vzorce a trendy v chování obyvatel a firem v oblasti odpadového hospodářství. Dají se získat užitečné informace, které lze použít pro různé účely a pro podporu rozhodování ať již při strategickém plánování nebo při podpoře inovací v této oblasti. I prostým sledováním vývoje po nějaké časové období je možné cílit motivaci k pokroku či nalézt příčiny negativního směřování.

Ačkoli je k dispozici množství nástrojů pro statistické zpracování dat, ne vždy je jednoduché s nimi pracovat a pro hodnocení dat z odpadového hospodářství nejsou přímo připraveny. To jsou důvody, které vedly ke vzniku této bakalářské práce.

Aktuálnost této problematiky je potvrzena i tvorbou nového ISOH (Informační systém odpadového hospodářství), který spravuje CENIA a Ministerstvo životního prostředí. Případně probíhajícím výzkumným projektem CEVOOH, který se na problematiku evidence dat v odpadovém hospodářství zaměřuje. V souvislosti s řešením projektu CEVOOH realizovaného na Ústavu procesního inženýrství na Vysokém učení technickém v Brně byla poskytnuta data pro vypracování této práce.

Bakalářská práce si klade za cíl vytvořit výpočtový nástroj včetně uživatelského rozhraní pro statistické zpracování dat z odpadového hospodářství. Spolupracujícím pracovištěm byla poskytnuta data z ročního hlášení o produkci a nakládání s odpady v detailu ZÚJ pro dva druhy odpadu: Oděvy (katalogové číslo 200110) a Textilní materiály (katalogové číslo 200111). Snahou bylo na základě těchto dat připravit takové řešení, které by bylo nezávislé na konkrétních druzích odpadu a roku, a bylo obecně použitelné pro celou škálu odpadů.

V teoretické části se práce dotýká oblasti popisné statistiky a zjišťuje postupy pro výpočet charakteristik polohy výběrového souboru a zpracování grafů, které jsou použity v praktické části práce. Pro zpracování výpočtového modulu byl zvolen programovací jazyk Python, proto je další kapitola teoretické části věnována právě jemu. Byl popsán rovněž obecný proces analýzy dat, který je uplatňován v praktické části. Poslední kapitola je věnována odpadovému hospodářství, kde je popsán katalog odpadů a kódování nakládání, a je nezbytným teoretickým základem pro porozumění zdrojovým datům.

Popisem poskytnutých zdrojových dat je zahájena praktická část práce. Na tu navazuje představení konceptu tvorby modelu, z něhož vychází zpracování dalších kapitol. Je popsána část načtení a čištění dat. Poté se již práce věnuje tvorbě grafického uživatelského rozhraní (GUI), ve kterém se provádějí uživatelské výběry podmnožiny dat, na něž se aplikují výpočtové funkce. GUI je také prostředek pro zobrazení zpracovaných dat a grafů. Stěžejní částí této kapitoly je popis tvorby funkcionalit pro statistické zhodnocení dat.

Ač je práce rozsáhlá, byly nalezeny omezení a limity zvoleného přístupu, které však již přesahují rozsah bakalářské práce. Rovněž byly vytyčeny oblasti, o které by se práce mohla v budoucnu dále obohatit. To vše je popsáno v poslední kapitole.

Přínosy práce spočívají v praktickém vytvoření výpočtového modulu s GUI, který je postaven pro řešení statistického zpracování dat zejména metodami popisné statistiky. Výpočtové funkce jsou aplikovány na uživatelem zvolenou podmnožinu dat základního souboru. Výsledky jsou zobrazovány v textovém widgetu GUI nebo v grafech či mapách.

I. TEORETICKÁ ČÁST

1 POPISNÁ STATISTIKA

Jeden ze způsobů, kterým lze pohlížet na statistiku je, že se jedná o praktickou činnost spočívající ve sběru, zpracování a vyhodnocování statistických údajů. Číselné údaje o jednotlivých skutečnostech však netvoří ještě statistiku. Aby se dali vyslovit závěry o zkoumaných jevech je potřeba provádět hromadné pozorování, jehož výsledkem je hromadný jev. [1]

Disciplína, která popisuje a sumarizuje informace obsažené ve velkém množství dat pomocí tabulek, grafů, funkcionálních a číselných charakteristik se nazývá popisná statistika. [2]

Statistické zkoumání se zabývá hromadnými jevy a procesy, jež se vyskytují u velkého množství prvků. Tyto prvky se nazývají statistickými jednotkami a považují se za elementární jednotky statistického pozorování. Vlastnosti statistických jednotek vyjadřují statistické znaky. Pokud bychom statistické znaky dělili podle toho, jak lze vyjádřit jejich obměny, dostaneme tyto skupiny znaků:

- kvantitativní (obměny znaku lze vyjádřit číselně)
    - měřitelné (hodnoty se dají porovnávat rozdílem nebo poměrem)
        - spojité (v rámci určitého intervalu mohou nabývat libovolných hodnot)
        - nespojité (mohou nabývat jen některých číselných hodnot)
    - pořadové (odrážejí pouze pořadí jednotlivých jednotek)
- kvalitativní (znaky vyjádřené slovně)
- alternativní (statistický znak může nabývat pouze dvou variant)
- množné (znaky, které připouštějí více než dvě varianty)

[1]

Hromadné jevy jakožto výsledky hromadného pozorování je možné pořizovat zejména dvěma způsoby:

- jako výsledky hromadných pokusů a
- jako výsledky pozorované na velkém počtu jednotek [3].

1.1 Základní a výběrový soubor

Předmětem statistického zájmu je soubor objektů, které tvoří základní soubor. Rozsah základního souboru může být konečný i nekonečný, obvykle je však velký, a proto je pozornost při vyšetřování soustředěna jenom na určitý omezený počet objektů, který se pak považuje za výběrový soubor. Výsledky získané z výběrového souboru je pak možno použít k provádění úsudku o základním souboru. [1], [2]

V této bakalářské práci se za základní soubor považují data pořízená prostřednictvím statistického šetření nesoucího podobu výkaznictví. Data jsou shromažďována do pracovní databáze ISOH. Prakticky se jedná o výkazy jednotlivých ZÚJ (základní územní jednotka České republiky) o nakládání s odpady.

Za statistické jednotky v daném souboru budou považovány jednotky ZÚJ, které jsou určeny svým číslem a typem subjektu. Jednoznačné vymezení statistických jednotek z hlediska věcného, prostorového a časového bude v aplikaci určeno dle voleb uživatele a jeho cílů statistického zkoumání.

1.2 Charakteristiky polohy

Cílem popisné statistiky je zpřehlednit informace obsažené v datovém souboru. Pokud bychom potřebovali z datové množiny získat další informace v koncentrované formě, je to možné udělat pomocí určitých charakteristik. Charakteristiky polohy popisují úroveň jevu vyjádřeného číselným znakem a měří obecnou velikost hodnot znaku v souboru. Můžeme je rozdělit na průměry, které jsou počítány ze všech dat a ostatní míry polohy počítané z vybraných hodnot. [3]

K popisu rozdělení se používá velké množství charakteristik jako jsou například medián a modus. [4]

1.2.1 Aritmetický průměr

Aritmetický průměr je zcela převažujícím druhem průměru, který najde své použití v mnoha úlohách statistiky. Považuje se za těžiště dat, kdy součet podprůměrných hodnot je stejný jako součet nadprůměrných hodnot. Pokud se v souboru vyskytují vybočující hodnoty, bývá jimi silně ovlivněn. Vypočítá se z neuspořádaných zjištěných hodnot x. [1]

$x_1, x_2, \dots x_n$	zjištěné hodnoty
n	celkový počet pozorování

$\bar{x}$ aritmetický průměr (x s pruhem) [1]

$$\bar{x} = \frac{x_1 + x_2 + \ldots + x_n}{n} = \frac{\sum_{i=1}^{n} x_i}{n}$$

Rovnice 1. Výpočet průměru [1]

Pokud bychom slovně chtěli popsali výpočet aritmetického průměru, mohli bychom říci, že se jedná o součet všech hodnot v daném souboru dat dělený počtem těchto hodnot. Průměr nám dá informaci o střední hodnotě datového souboru, což vede k lepšímu porozumění vlastnostem souboru dat. [1]

1.2.2 Medián

Slovo medián je synonymem ke slovu „střed“ a jeho pojmenování již naznačuje, že je to číslo, které leží uprostřed souboru. Představuje tedy hodnotu, kdy je počet prvků s větší nebo stejnou hodnotou jako medián roven počtu prvků s menší nebo stejnou hodnotou jako medián (v případě, že je počet prvků lichý). [5]

1.2.3 Percentily

Percentily v normálním rozložení slouží k tomu, že rozdělí množinu čísel na 100 intervalů, kde každý z nich obsahuje 1 % prvků v množině. Percentily představují hranice (99 možných), kde se těchto 100 intervalů setkává. [5]

„Kvantil proměnné x, který odděluje 100.p% malých hodnot proměnné x od (1-p).100% velkých hodnot proměnné x, označujeme jako $\tilde{x}_{p.100}$ a nazýváme ho 100.p % kvantilem proměnné x.“ [1]

p je relativní četnost malých hodnot

Například $\tilde{x}_{50}$ je 50procentní kvantil, který rozděluje statistický soubor na dvě stejně četné poloviny a nazývá se medián. [1]

1.2.4 Kvartily

Množiny dat lze rozdělit i jiným způsobem nežli pomocí percentilů. Kvartil například dělí množinu dat na čtyři intervaly, kde každý z nich obsahuje 25 % prvků v množině. Kvartily můžeme nazývat jako první, druhý nebo třetí kvartil. [5]

1.2.5 Směrodatná odchylka

Směrodatná odchylka je často používanou mírou statistické proměnlivosti. Na rozdíl od rozptylu je vyjádřena ve stejných jednotkách jako sledovaný znak. [3] Pokud tedy v bakalářské práci budu zjišťovat směrodatnou odchylku produkce odpadu na obyvatele, bude směrodatná odchylka pro uživatele lépe srozumitelná než rozptyl, který by byl ze své podstaty uváděn v jednotkách (odpad/obyvatele)^2.

Povahu rozložení můžeme popsat pomocí rozptylu, který udává, jak jsou hodnoty rozptýleny. Vypočítá se jako průměr druhých mocnin vzdáleností od průměru (Rovnice 2). [5]

Rozptyl má tvar:

$$s_x^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n}$$

Rovnice 2. Rozptyl [6]

Směrodatnou odchylku vypočítáme jako kladnou odmocninu z rozptylu podle vzorce (Rovnice 3):

$$s_x = \sqrt{s_x^2}$$

Rovnice 3. Směrodatná odchylka [3]

1.3 Četnosti

Prvky základního souboru, které vykazují určitou společnou vlastnost, tvoří množinu. Při analýze dat se pak zkoumá absolutní a relativní četnost prvků této množiny v daném výběrovém souboru. [2]

Absolutní četnost je informace o tom, kolikrát byla každá jedna obměna znaku obsažena v souboru. Relativní četnost říká, jaký je podíl výskytů konkrétní obměny na celkovém počtu zjištěných hodnot. [3]

1.3.1 Histogram

Pokud potřebujeme posoudit charakter dat nejprve vizuálně, histogram je vhodným nástrojem pro získání základní představy o rozdělení jednotlivých proměnných. [7] Pro grafické vyjádření poměrových a intervalových dat je histogram zřejmě nejpoužívanějším grafickým nástrojem. Histogram je vytvářen jako sloupcový graf s tím, že základny jednotlivých

sloupců mají délku zvolených intervalů a výšky sloupců mají velikost příslušných třídních četností. [1], [8]

Pro určení optimálního počtu (k) a šířky intervalů (h) existuje několik pravidel např.

- Sturgesovo pravidlo $k \approx 1 + 3{,}32 log_n$
- Yuleovo pravidlo $k \approx 2{,}5\sqrt[4]{n}$
- a další pravidla např. $k \approx \sqrt{n}$, nebo $k \approx 5 log_n$. [3]

Podle vzorců se stanoví vhodné k a stanoví se šířka intervalů ze vztahu (Rovnice 4):

$$h = \frac{R}{k}$$

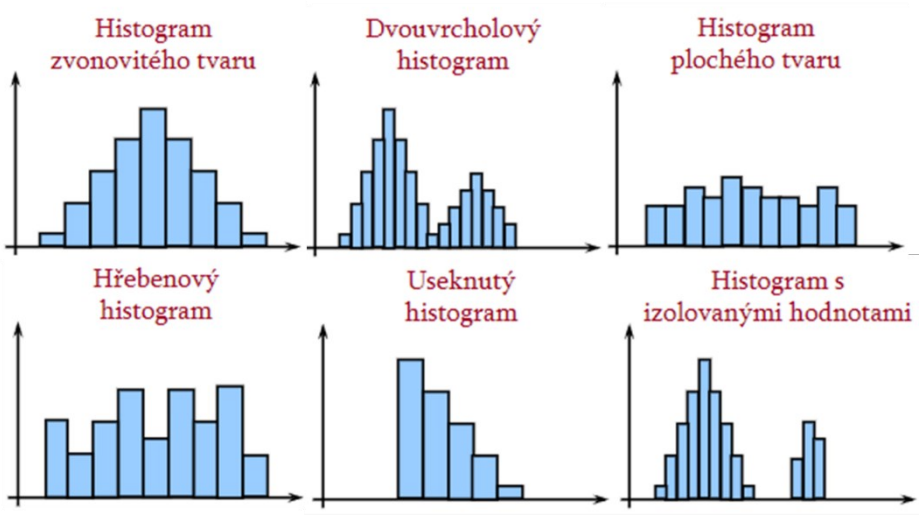
Rovnice 4. Stanovení šířky intervalu [3]

Variační rozpětí R zjistíme tak, že uspořádáme data podle velikosti a zjistíme nejmenší a nejvyšší hodnotu sledovaného znaku a tyto hodnoty od sebe odečteme (Rovnice 5). [3]

$$R = X_{max} - X_{min}$$

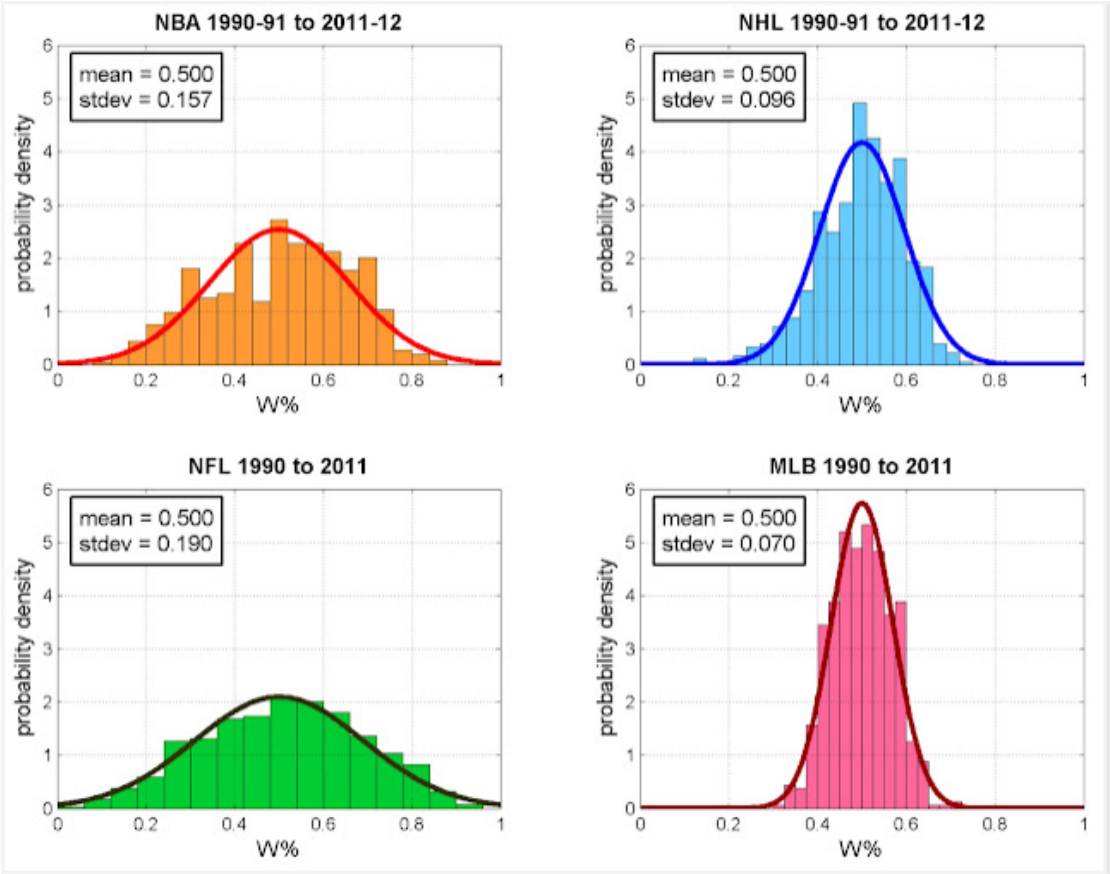
Rovnice 5. Variační rozpětí [3]

Na obrázku (Obrázek 1) jsou znázorněny typické tvary histogramů. Pokud porovnáme histogramy mezi sebou lze z jejich tvaru vyčíst možné působení vymezitelných jevů. Obrazem normálního rozdělení je histogram zvonovitého tvaru. Působí zde pouze náhodné příčiny variability. Pokud má histogram dva vrcholy, působí navíc i vymezitelné příčiny variability, kterými může být například spojení dvou souborů získaných za různých podmínek. Pokud je histogram plochého tvaru, může to být způsobeno výsledkem součtu několika rozdělení. U hřebenového histogramu působí i vymezitelné příčiny variability, kterými mohou být: nesprávné zaokrouhlování hodnot, chyby při zařazování do tříd či chyby v samotném měření hodnot. Useknutý histogram může poukazovat na omezenou rozlišovací schopnost. Histogram s izolovanými hodnotami naznačuje, že mohlo dojít k chybám při přepisování údajů či k chybám při měření. [9]



Obrázek 1. Tvary histogramů [9]

Jak již bylo zmíněno směrodatná odchylka je mírou variability datového souboru. Vliv směrodatné odchylky na tvar histogramu můžeme pozorovat na obrázku (Obrázek 2).



Obrázek 2. Vliv směrodatné odchylky na tvar histogramu [10]

Pokud je hodnota směrodatné odchylky malá, histogram bude mít strmější tvar. Pokud je směrodatná odchylka velká, hodnoty v souboru jsou rozptýlené kolem průměru a histogram bude mít širší tvar. [10]

Autor Hošek [10] uváděl demonstraci tvarů histogramů na výsledcích v nejvyšších sportovních amerických soutěžích (Obrázek 2). Popisuje, že histogramy jsou sestaveny z relativních počtů výher týmů v soutěžích basketbalu, hokeje, amerického fotbalu a baseballu. Širší a nižší histogram znamenají, že v soutěži NFL a NBA panují mezi jednotlivými týmy větší rozdíly. Vyvozuje, že zde existují trvale dominující týmy a týmy s nízkou výkonností. Kdežto úzké a vysoké grafy naznačují vyrovnanost soutěže. [10]

1.4 Krabicové diagramy

Krabicový diagram slouží jako nástroj pro grafické zobrazení ukazatelů polohy. Pomocí krabicového diagramu jsme schopni na jednom místě současně znázornit medián, hodnoty dolního a horního kvartilu, identifikovat odlehlé hodnoty a posoudit symetrii v okolí kvantilů a u konců rozdělení. [11]

Graf sestrojíme tak, že nejprve vypočteme délku obdélníku, která je rovna kvartilovému rozpětí, které určíme ze vztahu (Rovnice 6):

$$R = x_{75} - x_{25}$$

Rovnice 6. Kvartilové rozpětí [11]

Hodnota konce paprsků, někdy též nazývaná tykadla, je stanovena tak, že se od hodnoty kvartilu odečte jeden a půl násobek kvartilového rozpětí (Rovnice 7):

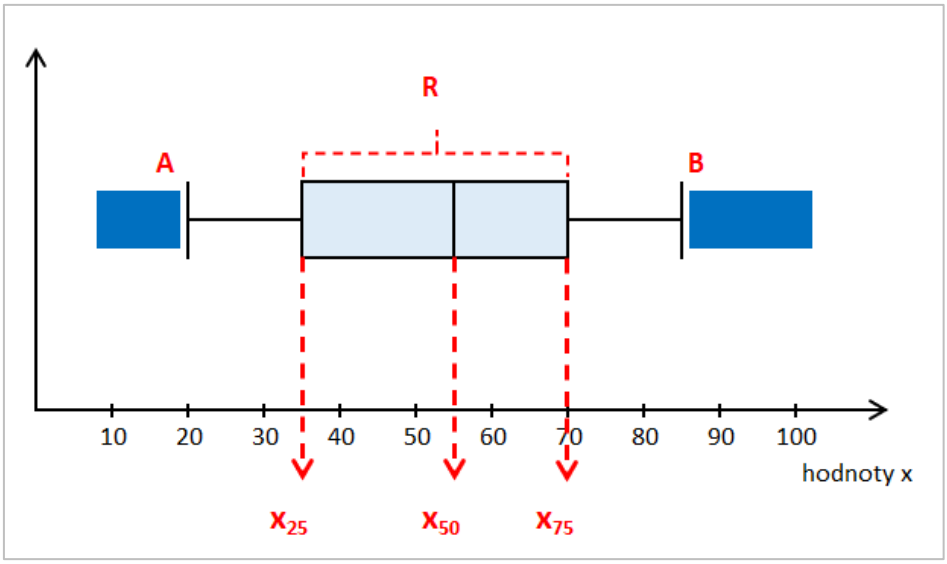
[11]

$$A = x_{25} - 1{,}5\,R$$

$$B = x_{75} + 1{,}5\,R$$

Rovnice 7. Určení konce paprsků [11]

Na obrázku (Obrázek 3) je možné shlédnout jednotlivé části krabicového diagramu. Přilehlé hodnoty A, B jsou konce paprsků, za kterými se na obou stranách nacházejí oblasti s odlehlými hodnotami. [11] Např. krabicový diagram, který je k dispozici v knihovně Matplotlib boxplot() zobrazuje odlehlé hodnoty jako kroužky.



Obrázek 3. Krabicový diagram [11]

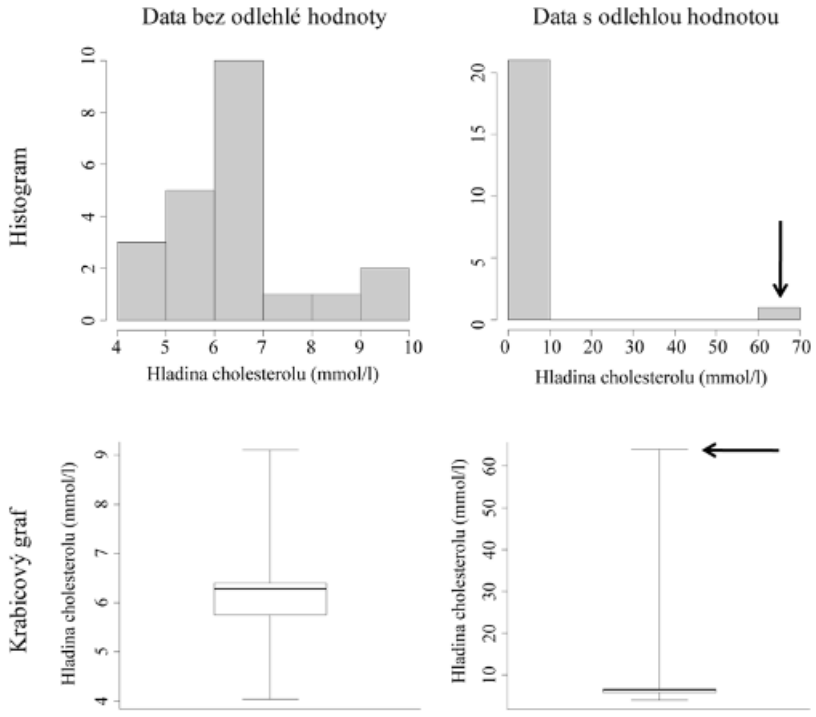
Tvar rozdělení dat popisuje šikmost a špičatost. Případy asymetrie a koncentrace rozdělení kolem středu jsou znázorněny na tvaru histogramu a krabicového diagramu a jsou uvedeny v příloze (P VI), grafy zpracovaly autorky [12].

1.5 Odlehlé hodnoty

Odlehlé hodnoty často bývají výsledkem chybného měření či překlepu, ale může se také stát, že nějaká odlehlá hodnota je hodnotou správnou. Od ostatních hodnot se liší výrazně díky tomu, že například vznikla při výjimečné situaci. Ačkoliv mohou být tyto hodnoty správné, je vhodné je dočasně vyloučit z analýzy a zpracovávat je odděleně. [13]

Definice extrémních (odlehlých) hodnot vždy závisí na oboru hodnot a konkrétním problému. V díle [8] autor Jarkovský uvádí, že: „někteří autoři definují odlehlou hodnotu jako hodnotu, která leží několikanásobek (tří, pěti, sedminásobek) výběrové směrodatné odchylky, respektive kvartilového rozpětí (často jedna a půl nebo třínásobek IQR), od průměru, respektive mediánu.“

Vhodnými nástroji pro rozpoznání, zda datový soubor obsahuje odlehlé hodnoty, jsou histogram a krabicový diagram. Pomocí zmiňovaných grafů (Obrázek 4) téměř vždy jednoznačně odhalíme problematickou hodnotu a poznáme ji tak, že je nezvykle vzdálená od ostatních pozorovaných hodnot. [8]



Obrázek 4. Identifikace odlehlé hodnoty pomocí histogramu a krabicového grafu [8]

2 PYTHON

Podle PYPL PopularitY of Programming Language indexu je Python celosvětově nejpopulárnějším programovacím jazykem. (Měřítkem je četnost vyhledávání návodů na Google.) [14] Pro práci s velkými daty je Python ideálním nástrojem. Díky bohatému ekosystému knihoven a výkonných balíčků je možné pomocí Pythonu provádět všechny druhy operací od běžného zpracování dat, vizualizace a statistické analýzy až po nasazení modelů strojového a hlubokého učení. [15]

Jedním z významných nástrojů Pythonu pro manipulaci, zkoumání a vizualizaci dat je knihovna Pandas. Ta byla vyvinuta Wes McKinney v roce 2008, ale skutečné popularity se jí dostalo až po roce 2012. Dnes je Pandas vnímána jako nezbytná knihovna pro analýzu dat v Pythonu.[16]

Dalšími populárními balíčky jsou Numpy, Matplotlib či Seaborn.

- Matplotlib – modul, který obsahuje nástroje pro vytvářejí různých typů grafů. [16]
- Pandas – umožňuje načítání dat z různých zdrojů jako jsou například CSV, Excel, SQL databáze a další. Data jsou ukládána do datových struktur DataFrames a series. S daty je potom možno provádět různé operace jako je slučování, filtrování a agregace či transformace. [17]
- Seaborn – je knihovna jejíž základ tvoří modul Matplotlib. Umožňuje tvorbu pokročilejších vizualizací. [18]
- NumPy – je základní knihovnou, která poskytuje nástroje pro vědecké výpočty v Pythonu.
- A další. …

Široká uživatelská základna, knihovny a balíčky přímo určené pro analýzu dat, to jsou důvody, proč byl pro zpracování bakalářské práce vybrán Python jako programovací jazyk.

2.1 Datové struktury v Pandas

- **DataFrame**

V Pandas se velmi často pracuje s datovou strukturou DataFrame. Jedná se o dvou dimenzionální tabulku, která může zvětšovat svou velikost. DataFrame strukturuje svá data do sloupců, a každý z těchto sloupců může mít různý datový typ. DataFrame obsahuje ozna-

čení řádků i sloupců a při matematických operacích se na řádky a sloupce můžeme odkazovat.

***class* pandas.DataFrame(*data=None*, *index=None*, *columns=None*, *dtype=None*, *copy=None*)**

[19], [20]

- **Series**

Sérii si lze představit jako tabulku, která má jen jeden sloupec. Je to jednorozměrné pole obsahující data libovolného typu. [21]

- **Geo Data Frames**

Pro práci s geoprostorovými daty poskytuje knihovna Pandas speciální balíček Geopandas. Pomocí tohoto rozšíření je možné vytvořit datové struktury Geo Data Frames. [22]

K vizualizaci geoprostorových dat se využívá zeměpisná délka a šířka. Zeměpisná délka jsou čáry, které vedou ze severu na jih a zeměpisná šířka jsou čáry, které vedou z východu na západ. Zeměpisná délka je vždy vykreslena podél vodorovné osy, zatímco zeměpisná šířka je vykreslena podél svislé osy grafu. Existují tři základní druhy geometrie: bod (jeden pár zeměpisné délky a šířky), čára (dva nebo více párů zeměpisné délky a šířky, které lze spojit tak, aby vytvořily souvislý segment) a mnohoúhelník – polygon (složený ze tří nebo více párů zeměpisných délek a šířek), které jsou spojeny v určeném pořadí, aby vytvořili uzavřenou oblast. [22]

GeoJSON je formát pro geoprostorová data vydaný organizací Internet Engineering Task Froce (IETF) v roce 2016 pod specifikací RFC 7946. Podporuje tyto typy geometrie: Point, LineString, Polygon, MultiLineString a MultiPolygon. [23]

Pokud načteme GeoJSON prostřednictvím Geopandas do GeoDataFrame, tak se vlastnosti souboru (názvy a geometrie) načtou jako jednotlivé sloupce. [22]

2.2 Uživatelské prostředí Tkinter

V Pythonu existuje několik frameworků, ve kterých je možno vyvíjet GUI – grafické uživatelské prostředí. Mezi nejpoužívanější patří: PyQT, Kivy, Jython WxPython, PyGUI a Tkinter. [24] Tkinter umožňuje v Pythonu vytvářet okna podobné těm z operačního systému Windows. [25]

Tkinter je známý svou jednoduchostí a dodává se společně s instalací Python. [25] Měl by fungovat stejně na různých operačních systémech, proto byl v bakalářské práci využit při tvorbě GUI aplikace.

Vytváření GUI je provázeno těmito fázemi:

- Import klíčové komponenty Tkinter modulu.
- Inicializace správce oken prostřednictvím metody tkinter.Tk()
- Dále je možné nastavit název okna pomocí window.title().
- Použití widgetů.
- Použití geometry managementu k zobrazení widgetu metody pack() případně grid()
- Posledním krokem je použití metody mainloop() k zobrazení okna.

[24]

3 PROCES ANALÝZY DAT

Analýza dat je chápána jako proces shromažďování, modelování a analýzy dat pomocí různých statistických a logických metod a technik. Výsledky dávají možnost rozhodovat se na základě faktů a nikoli prosté intuice. Pomáhají odhalit příležitosti k růstu či řešit neobvyklé situace dříve, než problémy nastanou. [26], [27]

Proces analýzy je popisován v pěti klíčových fázích:

- **Identifikace**

Nejprve je potřeba si ujasnit otázky, na které chceme znát odpověď. [27]

- **Sběr**

Fáze, ve které dochází ke sběru dat. Je potřeba definovat, které zdroje dat se budou používat a jakou bude mít sběr formu.

- **Čištění dat**

Ne všechna nasbíraná data budou užitečná. Je důležité se vyhnout poškození analýzy nekvalitními daty. Proto bude potřebné data vyčistit, odstranit nepotřebné záznamy, najít duplicity, odhalit bílá místa bez hodnot, vyřešit chyby s formátováním. [19], [27]

Existují postupy, jak se vypořádat s chybějícími hodnotami:

- Ověřit proč údaje chybí a pokud je to možné, zajistit jejich doplnění.
- Nahradit chybějící hodnoty jinými hodnotami.
- Odstranit všechny řádky s chybějícími daty z datového setu.
- Vyčlenit řádky s chybějícími daty do odděleného datasetu a pracovat s nimi zvlášť.

[19]

Protože vyřazením některých řádků může dojít ke zkreslení výsledků analýzy, je možnost doplnění chybějících údajů brána jako primární. Pokud Pandas narazí na prázdnou buňku, vloží místo ní do tabulky speciální hodnotu NaN. [16]

- **Analýza dat**

Pomocí různých technik statistické analýzy, regrese, textové analýzy a dalších je možné analyzovat a manipulovat s daty za účelem získání relevantních závěrů. V této fázi se odhalí trendy, korelace, variace a vzory a najdou se odpovědi na otázky položené v první fázi. [16], [27]

- **Interpretace**

Jedním z nejdůležitějších kroků je interpretace výsledků. V této fázi se na základě zjištění přichází s postupy a mohou se najít i různá omezení, na kterých se dá následně pracovat.

[26], [27]

4 ODPADOVÉ HOSPODÁŘSTVÍ

V České republice se odpadové hospodářství řídí podle zákona č. 541/2020 Sb. o odpadech. Dle tohoto zákona a vyhlášky č. 273/2021 Sb. o podrobnostech týkajících nakládání s odpady je zpracováváno Roční hlášení od původců a oprávněných osob, které je zasíláno do Integrovaného systému plnění ohlašovacích povinností (ISOP). [28]

4.1 Katalog odpadů

Vyhláška č. 8/2021 Sb. stanovuje mimo jiné Katalog odpadů a postup pro zařazování odpadu podle tohoto katalogu. Katalog odpadů je součástí přílohy zmíněné vyhlášky.

Odpady jsou zařazovány pod šestimístná katalogová čísla druhů odpadů. První dvojčíslí označuje skupinu odpadů, druhé dvojčíslí podskupinu odpadů a třetí dvojčíslí druh odpadu. Obecný postup zařazování je stanoven tak, že nejprve se vyhledá odpovídající skupina podle odvětví, oboru nebo technologického procesu, v němž odpad vzniká. Uvnitř skupiny se potom určí podskupina odpadu. V dané skupině se následně vyhledá název druhu odpadu s příslušným katalogovým číslem. [29]

Katalog odpadů obsahuje:

- 20 skupin odpadů,
- 111 podskupin odpadů a
- 839 druhů odpadů.

Spolupracujícím pracovištěm byla předána data pro dva druhy odpadů 200111 a 200110. Dle Katalogu odpadů se jedná o odpady ze skupiny 20 a podskupiny 20 01 čili:

> **20** Komunální odpady (odpady z domácností a podobné živnostenské, průmyslové odpady a odpady z úřadů), včetně složek z odděleného sběru
>
> **20 01** Složky z odděleného sběru (kromě odpadů uvedených v podskupině 15 01)
>
> 20 01 11 Textilní materiály
> 20 01 10 Oděvy
>
> [29]

Tyto druhy byly zvoleny záměrně, protože je u nich předpoklad, že zde budou patrny rozdíly v nakládání v rámci České republiky. [29]

V příloze práce (P I [29]). pro názornost uvádím Katalog odpadů. Z důvodu rozsáhlosti katalogu pouze pro skupinu 20.

4.2 Kódování nakládání

Kódování nakládání s odpady se provádí dle tabulky Kódy původu odpadu a způsobů nakládání s odpady pro evidenční účely, která je uvedena v příloze (P II [30]). Tato tabulka je rozdělena na dvě hlavní části:

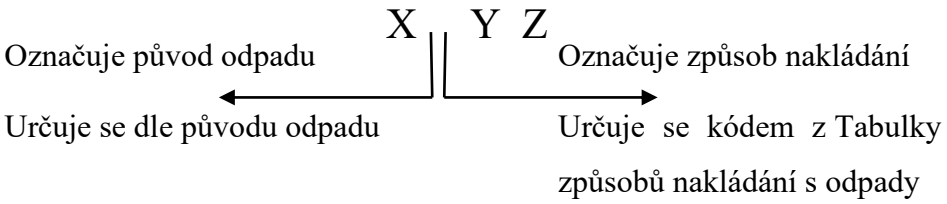
- Původ odpadů
- Způsob nakládání s odpady
    - Využívání odpadů
    - Odstraňování odpadů
    - Ostatní.

Tabulka má čtyři sloupce (Obrázek 5). První sloupec uvádí popis činnosti a ve druhém najdeme kódové označení. Následuje sloupec Množství odpadu +/-. V tomto sloupci je stanoveno, zda daná činnost zvyšuje či snižuje množství odpadu v ročním hlášení. Poslední sloupec uvádí, zda je potřeba zaznamenat partnera nebo ne. U těch kódů nakládání, kde partner vyžadován není, je v tabulce se zdrojovými daty uveden jako partner evident. [30], [31]

KÓDOVÁNÍ ZPŮSOBŮ NAKLÁDÁNÍ S ODPADY			
PŮVOD ODPADŮ			
Původ odpadů	**Kód**	**Množství odpadu +/-**	**Partner**
Produkce odpadu (vlastní vyprodukovaný odpad)	A00	(+)	NE
Odpad převzatý od původce, jiné oprávněné osoby (sběr, výkup, shromažďování), nebo jiné provozovny	B00	(+)	ANO
Množství odpadu převedené z minulého roku (zůstatek na skladu k 1. lednu vykazovaného roku)	C00	(+)	NE
ZPŮSOB NAKLÁDÁNÍ S ODPADY			
Využívání odpadů			
Způsob	**kód**	**+/-**	**partner**
Využití odpadu způsobem obdobným jako paliva nebo jiným způsobem k výrobě energie	XR1	(-)	NE
Získání /regenerace rozpouštědel	XR2	(-)	NE
Získání/regenerace organických látek, které se nepoužívají jako rozpouštědla (včetně biologických procesů mimo kompostování a biologickou dekontaminaci)	XR3	(-)	NE

Obrázek 5. Ukázka záhlaví tabulky pro kódování způsobů nakládání s odpady [30], [31]

Způsob nakládání s odpady se označuje třímístným kódem XYZ (Obrázek 6). První písmeno X označuje původ odpadu. Druhé písmeno a číslice znamenají kód způsobu nakládání s odpadem. [30], [31]



Obrázek 6. Schéma struktury kódu pro nakládání s odpady [30], [31]

U původu odpadu (**písmeno X**) rozlišujeme tři možnosti:

- A00 – pro vlastní (vyprodukovaný) odpad
- B00 – pro odpad převzatý
- C00 – pro odpad odebraný ze zásob z předchozího roku.

Takže potom kód:

- AYZ znamená, že se jedná o vlastní odpad, s nímž bylo naloženo způsobem YZ
- BYZ určuje, že se jedná o převzatý odpad, se kterým bylo naloženo způsobem YZ
- CYZ říká, že se jedná o odpad odebraný ze zásob z předchozího roku, se kterým bylo naloženo způsobem CYZ.

Druhé **písmeno Y** může nabývat znaků: 0, R, D nebo N:

- R – využívání odpadů
- D – odstraňování odpadů
- N – ostatní způsoby.

A poslední **písmeno Z** může obsahovat 0 nebo jedno či dvouciferné číslo.

[30], [31]

II. PRAKTICKÁ ČÁST

5 POPIS ZDROJOVÝCH DAT

V rámci řešení projektu CEVOOH realizovaného na Ústavu procesního inženýrství Vysokého učení v Brně byly poskytnuty zdrojové soubory ve formátu CSV. Dílčí datová sada byla získána z pracovní databáze ISOH (zkráceně PD ISOH). Tato data byla vhodně upravena pro potřeby této bakalářské práce.

Data jsou strukturována v tabulce o jedenácti sloupcích a jedná se vlastně o popis vztahu mezi zúčtovacími jednotkami, tj. evidentem a partnerem (Obrázek 7). Pro každý druh odpadu jsou data uváděna v samostatném CSV souboru.

	A	B	C	D	E	F	G	H	I	J	K
1	Evident	Evident - Typ subjek	Evident - Název	Evident - Kraj	Partner	Partner - Typ subjektu	Partner - Název	Partner - Kraj	Kód	Indikátor	Množství
795	554782	1	Praha	Hlavní město F	554782	1	Praha	Hlavní město Pr	C00	Vyskladnění	xxx

Obrázek 7. Ukázka zdrojových dat před anonymizací

Z důvodu neveřejnosti dat byla provedena anonymizace, která spočívala ve změně čísla ZÚJ (základní územní jednotka) a názvu ZÚJ. Dále budou v práci prezentována již pouze data tímto způsobem anonymizovaná. Nebo bude zatajen údaj o množství odpadu.

Evident i partner jsou identifikováni svými identifikačními čísly, typem subjektu, názvem a krajem. Kromě geografických údajů jsou zásadní dvě informace. První se týká typu odpadu a je definována katalogovým číslem (4.1). Data pro jednotlivá katalogová čísla odpadů jsou poskytnuta vždy v samostatném CSV souboru. Druhá podstatná informace se týká způsobu nakládání s odpady (4.2) a je evidována pod sloupcem Kód v tabulce na obrázku (Obrázek 8). Ve sloupci Indikátor jsou jednotlivé kódy zařazeny do skupin dle podobnosti pro statistické vyhodnocování. Část tabulky je zobrazena níže na obrázku (Obrázek 8).

	B	C	E	F	H	I	K	L	M	N	O
1	Evident _A	Evident - Typ subjek	Evident - Nazev_A	Evident - Kraj	Partner _A	Partner - Typ subjektu	Partner-Nazev_A	Partner - Kraj	Kód	Indikátor	Množství
2	41269	1	KIZBUB	Hlavní město F	41269	1	KIZBUB	Hlavní město Pr	A00	Produkce	39,11
3	40003	4	HAFHOT	Zlínský	40003	4	HAFHOT	Zlínský	A00	Produkce	4,731
4	40006	1	HAPHET	Olomoucký	40006	1	HAPHET	Olomoucký	A00	Produkce	297,116
5	41273	1	KIFFOB	Ústecký	41273	1	KIFFOB	Ústecký	A00	Produkce	2,215
6	40030	4	HATDAT	Olomoucký	40030	4	HATDAT	Olomoucký	A00	Produkce	3,166
7	40038	4	HAVDOT	Olomoucký	40038	4	HAVDOT	Olomoucký	A00	Produkce	2,997
8	40054	1	HALNUT	Zlínský	40054	1	HALNUT	Zlínský	A00	Produkce	0,183
9	40055	1	HAMNAT	Moravskoslezs	40055	1	HAMNAT	Moravskoslezsk	A00	Produkce	33,40289319
10	40071	4	HANFET	Jihočeský	40071	4	HANFET	Jihočeský	A00	Produkce	3,5312
11	40089	1	HAZLUT	Olomoucký	40089	1	HAZLUT	Olomoucký	A00	Produkce	6,2855
12	40115	4	HEMKAN	Olomoucký	40115	4	HEMKAN	Olomoucký	A00	Produkce	6,014
13	40123	1	HEFRON	Olomoucký	40123	1	HEFRON	Olomoucký	A00	Produkce	0,02
14	40125	4	HEMRAN	Olomoucký	40125	4	HEMRAN	Olomoucký	A00	Produkce	5,912
15	40127	1	HESRIN	Olomoucký	40127	1	HESRIN	Olomoucký	A00	Produkce	0,025

Obrázek 8. Ukázka zdrojových dat pro katalogové číslo 200111 – Textilní materiály

Evident i partner jsou charakterizováni svým číslem, názvem a zařazením do územního celku ORP a kraje. U evidenta i partnera se rozlišuje typ subjektu. Nejčastěji vyskytující se typy subjektů v poskytnutých datech jsou 1 a 4. Jednička znamená že se jedná o firmu, čtverka označuje obec. Pro potřeby této práce vysvětlení tohoto specifika postačuje v tomto detailu.

Vztah mezi evidentem a partnerem je dán kódem nakládání, který je blíže popsán v kapitole 4.2. Poskytnutá data jsou agregována na úroveň ZÚJ. V rámci jednoho druhu odpadu musí být u konkrétního evidenta (popř. ZÚJ) v rámci ročního hlášení vyrovnány příjmové a výdajové operace (způsoby nakládání s odpady).

Pro lepší představu jsou uvedeny příklady, jak je možné číst záznamy v tabulce:

- Kód XR1, kterému byl přiřazen identifikátor Energetický říká, že evident 42355 ROMNAF náležející do Jihomoravského kraje využil 174,523 tun textilního odpadu kat. č. 200111 podobným způsobem jako palivo. Jako partner je uveden taktéž subjekt 42355 protože v rámci tohoto kódu nakládání není partner vyžadován (Obrázek 9).

	B	C	E	F	H	I	K	L	M	N	O
1	Evident_A	Evident_TypSubjektu	Evident_Nazev_A	Evident_Kraj	Partner_A	Partner_Typ_subjektu	Partner_Nazev_A	Partner_Kraj	Kod	Indikator	Mnozstvi
183	42355	1	ROMNAF	Jihomoravský	42355	1	ROMNAF	Jihomoravský	XR1	Energeticky	174,523

Obrázek 9. Záznam s kódem XR1

- Kód „Předání“ říká, že evident 41725 KIMRAS spadající do Ústeckého kraje předal partnerovi 41731 KINDES rovněž z Ústeckého kraje 3,22 tun textilního materiálu kat. č. 200111 (Obrázek 10).

	B	C	E	F	H	I	K	L	M	N	O
1	Evident_A	Evident - Typ subjek	Evident - Nazev_A	Evident - Kraj	Partner_A	Partner - Typ subjektu	Partner-Nazev_A	Partner - Kraj	Kód	Indikátor	Množství
450	41725	1	KIMRAS	Ústecký	41725	1	KIMRAS	Ústecký	A00	Produkce	3,22
1716	41725	1	KIMRAS	Ústecký	41731	1	KINDES	Ústecký	Předání	Předání	3,22

Obrázek 10. Záznam s kódem „Předání“, evident KIMRAS

A pokud se podíváme ve stejné tabulce na evidenta 41731 KINDES, tak toto předání nalezneme jako kód „Převzetí“ od partnera KIMRAS. Rovněž 3,22 tun (Obrázek 11).

	B	C	E	F	H	I	K	L	M	N	O
1	Evident _A	Evident - Typ subjek	Evident - Nazev_A	Evident - Kraj	Partner _A	Partner - Typ subjektu	Partner-Nazev_A	Partner - Kraj	Kód	Indikátor	Množství
453	41731	4	KINDES	Ústecký	41731	4	KINDES	Ústecký	A00	Produkce	3,92
842	41731	1	KINDES	Ústecký	41731	1	KINDES	Ústecký	C00	Vyskladnění	1,15
996	41731	1	KINDES	Ústecký	41731	1	KINDES	Ústecký	XR12	Materiálově	176,18
1053	41731	1	KINDES	Ústecký	41731	1	KINDES	Ústecký	XN5	Naskladnění	0,59
1719	41731	1	KINDES	Ústecký	41731	1	KINDES	Ústecký	Předání	Předání	5,81
1720	41731	4	KINDES	Ústecký	41742	1	KIBTIS	Ústecký	Předání	Předání	3,92
2564	41731	1	KINDES	Ústecký	41316	1	KOPKEF	Plzeňský	Převzetí	Převzetí	162,92
2784	41731	1	KINDES	Ústecký	41725	1	KIMRAS	Ústecký	Převzetí	Převzetí	3,22
2785	41731	1	KINDES	Ústecký	41727	1	KISRIS	Ústecký	Převzetí	Převzetí	1,1
2786	41731	1	KINDES	Ústecký	41729	1	KIZRUS	Ústecký	Převzetí	Převzetí	6,06
2787	41731	1	KINDES	Ústecký	41731	1	KINDES	Ústecký	Převzetí	Převzetí	5,81
2795	41731	1	KINDES	Ústecký	41742	1	KIBTIS	Ústecký	Převzetí	Převzetí	2,22
2806	41731	1	KINDES	Ústecký	41273	1	KIFFOB	Ústecký	Převzetí	Převzetí	0,1

Obrázek 11. Záznam s kódem „Převzetí“, evident KINDES

Ve zdrojové tabulce pak kód nakládání předurčuje, jaká matematická operace se použije při ročním zúčtování. Zda se bude množství odpadu přičítat, nebo odečítat. Pro každého evidenta by v ročním zúčtování měla vycházet nula (Obrázek 12). (Pozn.: Převzetí a Předání ve sloupci Kód je již agregováno z více kódů nakládání a bylo provedeno v rámci úprav pracovištěm na VUT Brno.)

	B	C	E	F	H	I	K	L	M	N	O	Q
1	Evident _A	Evident - Typ subjek	Evident - Nazev_A	Evident - Kraj	Partner _A	Partner - Typ subjektu	Partner-Nazev_A	Partner - Kraj	Kód	Indikátor	Množství	+/-
471	41782	1	KIBLIS	Vysočina	41782	1	KIBLIS	Vysočina	A00	Produkce	0,582	+
785	41782	1	KIBLIS	Vysočina	41782	1	KIBLIS	Vysočina	BN30	Produkce	2,61	+
1055	41782	1	KIBLIS	Vysočina	41782	1	KIBLIS	Vysočina	XN5	Naskladnění	3,192	+
1745	41782	1	KIBLIS	Vysočina	41782	1	KIBLIS	Vysočina	Předání	Předání	3,192	-
2813	41782	1	KIBLIS	Vysočina	41782	1	KIBLIS	Vysočina	Převzetí	Převzetí	3,192	-
3234											0	

Obrázek 12. Roční zúčtování pro evidenta KIBLIS

6 NÁVRH STATISTICKÉHO ZPRACOVÁNÍ DAT

Na obrázku (Obrázek 13) je představen návrh pro statistické zpracování poskytnutých dat. Jako programovací jazyk byl zvolen Python pro své knihovny Pandas, Matplotlib, které nabízí široké možnosti analýzy, vyhodnocení a vizualizace dat, včetně možnosti práce s geografickými podklady a možnosti připravit grafické uživatelské prostředí.

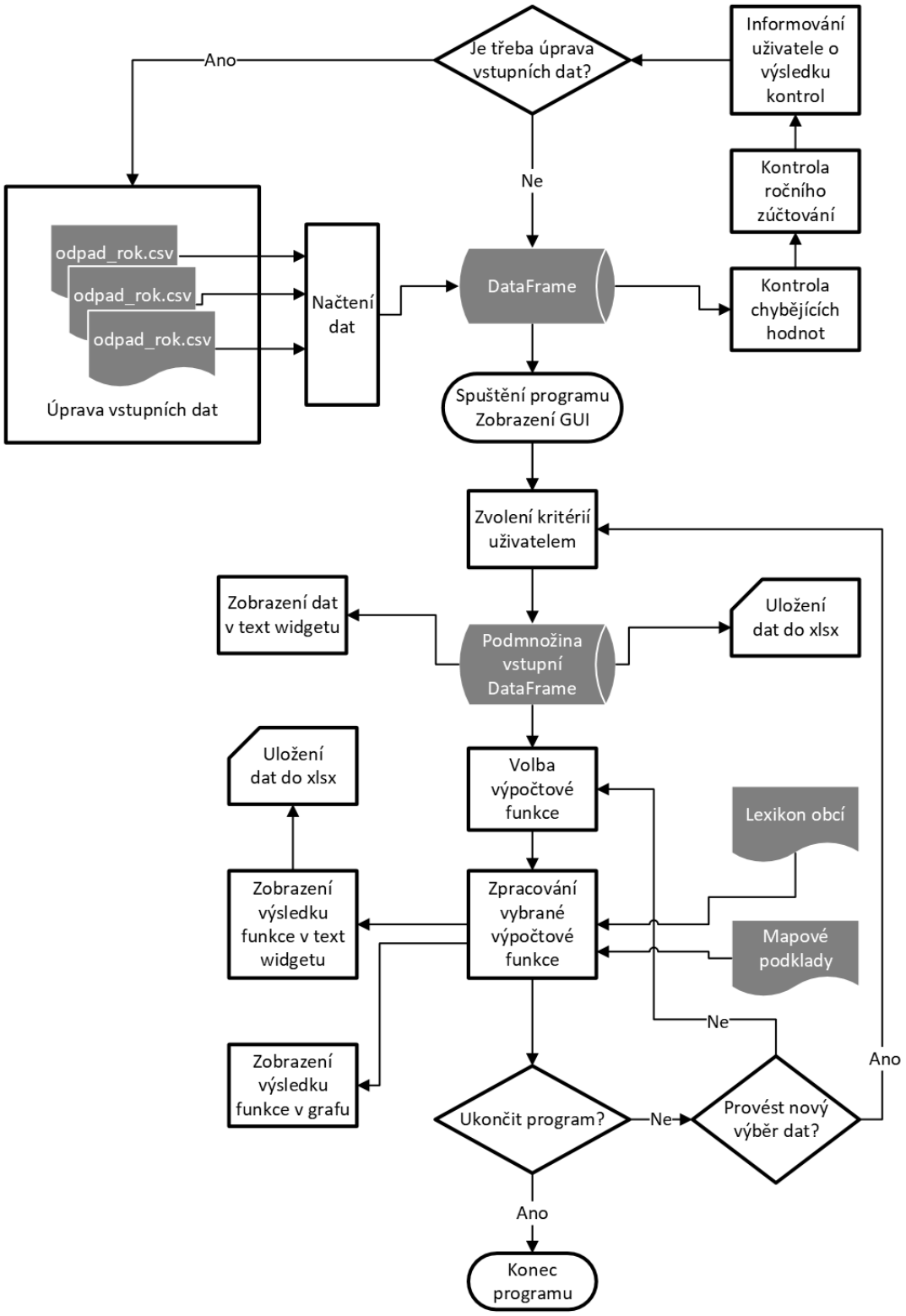
Cílem bakalářské práce je navrhnout, jakým způsobem by se dala data z ročního hlášení statisticky zpracovat. Aby byly výsledky navrhnutého řešení viditelné a ověřitelné, je připraveno uživatelské prostředí pomocí nástroje Tkinter v Pythonu.

Navržený koncept statistického zpracování ve svých hlavních bodech dodržuje obecnou metodiku pro analýzu dat popsanou v kapitole 3. Cílem je připravit model, který je univerzální a není závislý přímo na konkrétních datech odpadu 200111 (textilní materiály) a 200110 (oděvy). Z důvodu univerzálnosti je však stanoveno dodržování pravidel pro pojmenování a typ vstupních souborů včetně požadavku na jednotné názvy sloupečků v těchto souborech.

Data ze vstupních souborů se načítají do jedné tabulky DataFrame, ve které se provádí kontrola chybějících hodnot a kontrola nulového výsledku ročního zúčtování. Uživatel modelu je informován o výsledku kontrol a pokud jsou zjištěny nedostatky, upraví se data ve vstupních souborech.

Po spuštění programu je zobrazeno GUI (grafické uživatelské prostředí), ve kterém jsou uživatelem zadána kritéria výběru podmnožiny dat. Data jsou získána filtrováním vstupní DataFrame. Jednotlivé záznamy získané podmnožiny dat jsou zobrazeny v textovém widgetu. Tato data je možné uložit do nového souboru s příponou .xlsx.

Uživateli jsou nabídnuty připravené funkce pro statistické zpracování dat, které se spouštějí nad touto podmnožinou. Funkce nabízejí nástroje popisné statistiky. Pro zpracování map jsou načítány mapové podklady a některé funkce využívají Lexikon obcí. Výsledky funkcí jsou zobrazeny v textovém widgetu a v grafech, které se spouštějí v samostatných oknech. Výsledky funkcí je možno také ukládat do XLSX. Uživateli je umožněno spouštět jednotlivé funkce za sebou na zvolené podmnožině dat nebo provést nový uživatelský výběr dat a na závěr program ukončit.



Obrázek 13. Návrh postupu pro statistické zpracování zdrojových dat

7 ZPRACOVÁNÍ DAT

7.1 Načtení dat

Protože jsou názvy sloupečků zdrojových souborů využívány jako parametry funkcí a další tabulky se slučují podle čísel ZÚJ, stanovil se standard pojmenování názvů sloupečků. Tabulky se načítají do jedné Pandas DataFrame, ve které je odkazování na data realizováno právě přes názvy sloupců. Proto je nezbytné jejich přesné a standardizované pojmenování (Obrázek 14). Výhodou je, že pokud by například došlo ke změně pořadí sloupců zdrojové tabulky nebude to mít vliv na fungování skriptů.

Název sloupce	Hodnota
Evident_ZUJ_Cislo	500020
Evident_ZUJ_Nazev	Petrov nad Desnou
Evident_ORP_Nazev	Šumperk
Evident_ORP_Cislo	7111
Evident_Kraj_Nazev	Olomoucký kraj
Evident_Kraj_Cislo	CZ071
Evident_TypSubjektu	4
Partner_ZUJ_Cislo	500020
Partner_ZUJ_Nazev	Petrov nad Desnou
Partner_ORP_Nazev	Šumperk
Partner_ORP_Cislo	7111
Partner_Kraj_Nazev	Olomoucký kraj
Partner_Kraj_Cislo	CZ071
Partner_TypSubjektu	4
Kod	A00
Indikator	Produkce
Mnozstvi	x,xxx

Obrázek 14. Standardizace názvů sloupečků u zdrojových souborů

Načtení zdrojového souboru ve formátu CSV pomocí knihovny Pandas je pro účely bakalářské práce provedeno relativní cestou. Od uživatelů je pak pouze vyžadováno, aby zdrojové soubory umístili do složky s názvem Data, která se musí nacházet ve stejné adresářové složce, ve které se nachází spouštěný program, do kterého je soubor načítán.

Pro názvy souborů je stanovena rovněž přesná notace – číslo odpadu a rok. Například: „200110_2021“, což znamená odpad oděvy za rok 2021.

Pro načtení zdrojových dat (Program 1) je sestavena uživatelská funkce *load_csv_type_conversion()*, která využívá funkci modulu Pandas *pd.read_csv()*. Jelikož jsou data ve zdrojovém souboru oddělena středníkem, musí se středník uvést jako nepovinný parametr funkce. Aby se zajistilo, že se množství odpadu ze sloupce ´Mnozstvi´ správně načte a interpretuje se jako desetinné číslo, je potřeba knihovně Pandas specifikovat, který znak se ve zdrojovém souboru používá k oddělení desetinných míst. To se realizuje přes parametr decimal.

```
def load_csv_type_conversion(filename, dtypes):
    df = pd.read_csv(filename, delimiter=';', decimal=',',dtype =
dtypes)
    return df
```

Program 1. Funkce pro načtení dat z CSV do DataFrame [17]

Dále je zapotřebí, aby bylo pro některé sloupce jako např. Evident_ZUJ_Cislo, Evident_TypSubjektu explicitně určeno, jaký budou datový typ. Aby byla funkce pro načítání dat z CSV souborů znovu použitelná. Bude tato specifikace provedena přes slovník (Program 2). Slovník obsahuje klíče, které odkazují na názvy sloupců a hodnoty, které specifikují datový typ, který má být použit při načítání.

```
dtypes_lexikonObci ={
    'ZUJ_Cislo':      'string',
    'ZUJ_Nazev':      'string',
    'ORP_Cislo':      'string',
    'ORP_Nazev':      'string',
    'Kraj_Cislo':     'string',
    'Kraj_Nazev':     'string',
    'Pocet_Obyvatel': 'int'
```

Program 2. Slovník pro explicitní určení datových typů vstupních dat

Data se načítají do datové struktury DataFrame. Každý z názvů sloupců je formátu string, a proto se ve funkcích názvy sloupců volají v uvozovkách a jsou uzavřeny v hranatých závorkách. Při načítání dat se automaticky přidává nový sloupec s indexem. Index se počítá od nuly. Tzn. první načtený řádek má index 0.

Při načítání CSV souborů pro jednotlivé odpady je základní funkce rozšířena o další funkcionalitu (Program 3):

- Do jednoho DataFrame se prostřednictvím for cyklu načítají všechny soubory, které se nacházejí v adresářové složce (parametr: directory).

- Část z názvu souboru, která se nachází před podtržítkem, je přidána jako poslední sloupec do DataFrame (parametr: column_name). Je to z toho důvodu, aby se ve výsledné DataFrame mohl určit typ odpadu.
- Rok se nachází v druhé části názvu souboru za podtržítkem a ze stejného důvodu se také přidá jako poslední sloupec do DataFrame (parametr: column_year).

```
def load_files_to_df(directory,extension,dtypes,column_name,
    column_year):
    files = [file for file in os.listdir(directory) if
    file.endswith(extension)]

    df = pd.DataFrame()

    for file in files:
        filepath = os.path.join(directory, file)
        data = pd.read_csv(filepath,delimiter=';', decimal=',',
       ,usecols = None, dtype=dtypes)

        filename = os.path.splitext(file)[0]
        data[column_name] = filename.split("_")[0]
        data[column_year] = filename.split("_")[1]

        df = df.append(data)
        df[column_name] = df[column_name].astype(str)
        df[column_year] = df[column_year].astype(str)

    return df
```

Program 3. Rozšířená funkce pro načtení dat z CSV do DataFrame [17]

V praxi to tedy znamená, že všechny CSV soubory s jednotlivými druhy odpadu jsou umístěny do jedné složky, která je podsložkou adresáře, ve kterém je obsažen kód programu. CSV soubory jsou pojmenovány druhem odpadu a rokem např. 200111_2022.csv. Po provedení kódu jsou všechny soubory načteny do jednoho DataFrame a jsou přidány poslední dva sloupce s názvy zvolenými v parametru funkce *load_files_to_df()*.

Příklad volání funkce *load_files_to_df()* pro získání zdrojové DataFrame (Program 4).

```
Zdrojovy = load_files_to_df ('Data','.csv', dtypes_odpady,
           'Druh_Odpadu','Rok')
```

Program 4. Volání funkce *load_files_to_df()*

Obdobně je načten pomocí funkce *load_csv_type_conversion()* vstupní soubor s počty obyvatel České Republiky (Obrázek 15), lexikon obcí ČR a tabulka s kódy nakládání.

	A	B	C	D
1	Kod_Okresu	Kod_Obce	Nazev_Obce	Pocet_Obyvatel
11	CZ0201	529478	Čakov	129
12	CZ0201	529486	Čechtice	1390
13	CZ0201	529516	Čerčany	2874
14	CZ0201	529532	Červený Újezd	346
15	CZ0201	529541	Český Šternberk	171
16	CZ0201	529567	Čtyřkoly	714
17	CZ0201	532746	Děkanovice	60

Obrázek 15. Ukázka vstupního souboru

Pocet_obyvatel.csv

Lexikon obcí představuje soubor informací o obcích a městech České republiky (Obrázek 16).

	A	B	C	D	E	F
1	Evident_ZUJ_Cislo	Evident_ZUJ_Nazev	Evident_Kraj_Cislo	Evident_Kraj_Nazev	Evident_ORP_Cislo	Evident_ORP_Nazev
5	529451	Bystřice	CZ020	Středočeský kraj	2101	Benešov
6	529478	Čakov	CZ020	Středočeský kraj	2101	Benešov
7	529516	Čerčany	CZ020	Středočeský kraj	2101	Benešov
8	529541	Český Šternberk	CZ020	Středočeský kraj	2101	Benešov
9	529567	Čtyřkoly	CZ020	Středočeský kraj	2101	Benešov

Obrázek 16. Ukázka vstupního souboru LexikonObci.csv

7.2 Čištění dat

U DataFrame s načtenými daty o odpadech je potřeba provést kontrolu a případné čištění dat.

7.2.1 Kontrola chybějících hodnot

Kontrola, že v datasetu žádná data nechybí, je provedena pomocí funkce z knihovny Pandas *isnull()*. Tato funkce vrátí True pokud nalezne chybějící hodnotu. True má hodnotu 1, takže pokud bude následně výsledek sečten pomocí funkce *sum()*, získáme pro každý sloupec DataFrame počty chybějících hodnot (Program 5).

```
def je_soucet_nulovy(df):
    check_sum = df.isnull().sum().sum()  # součet všech NaN hodnot
v DataFrame
    if check_sum == 0:
        print('Nechybi zadna hodnota')
        return True
    else:
```

```
        missing_rows = df[df.isnull().any(axis=1)].index.tolist()
        # seznam řádků s chybějícími hodnotami
        print("Indexy radku s chybejicimi hodnotami: ",
        missing_rows)
        return False
```

Program 5. Funkce pro kontrolu existence řádků s chybějícími hodnotami

Pokud by se v datasetu vyskytla chybějící hodnota, bude se muset uživatel rozhodnout, zda doplní data ve vstupním souboru, odstraní neúplný záznam, nebo chybějící hodnotu nahradí nějakou jinou hodnotou. V každém případě to může způsobit komplikace při následné kontrole ročního zúčtování, protože ta samá hodnota bude chybět nebo přebývat v zúčtování na straně partnera.

7.2.2 Kontrola ročního zúčtování

Další kontrola je zavedena na úrovni ročního zúčtování, které by mělo pro každého evidenta v rámci jednoho druhu odpadu vyjít jako nulové (plusové a mínusové položky se vzájemně odečtou). Pro zdárné provedení tohoto výpočtu, je třeba ke zdrojovým datům doplnit informaci o tom, zda způsob nakládání s odpady zvyšuje či snižuje množství odpadu. Jedná se o sloupec ´Množství odpadu (+/-)´ v tabulce Kódování způsobů nakládání s odpady, která je uvedena v příloze (P II).

Přiřazení informace je provedeno pomocí funkce *merge()* z knihovny Pandas. Nejprve však dojde k úpravě zdrojové tabulky:

- Informace o kódování a původ či způsob zpracování se převedou na řádkové informace ke každému kódu zpracování.
- Přejmenování názvů sloupců
- Ze sloupce ´Množství odpadu (+/-)´ původní tabulky se získá informace o navýšení či úbytku (to je znak + či -) a pro snadnější následné výpočty se převede na hodnoty +1 a -1. Tyto hodnoty jsou vloženy jako nový sloupec s názvem ´Navyseni_Ubytek´. Při slučování tabulek se těmito hodnotami násobí množství odpadu v tabulkách s odpady (sloupec ´Mnozstvi´).

Náhled takto upravené tabulky je uveden na obrázku (Obrázek 17) a celá tabulka v příloze (P III). Tabulka je načítána jako DataFrame pod názvem Nakladani. Hodnoty ve sloupci ´Navyseni_Ubytek´ jako datový typ Int.

	A	B	C	D	E	F	G
1	Kod	Operator	Partner	Navyseni_Ubytek	Kodovani_zpusobu	Puvod_Zpusob	Popis_zpusobu
2	A00	(+)	NE	1	Původ odpadů	Původ odpadů	Produkce odpadu (vlastní vyprodukovaný odpad)
3	B00	(+)	ANO	1	Původ odpadů	Původ odpadů	Odpad převzatý od původce, jiné oprávněné osoby (sběr,
4	C00	(+)	NE	1	Původ odpadů	Původ odpadů	Množství odpadu převedené z minulého roku (zůstatek
5	XR1	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Využití odpadu způsobem obdobným jako paliva nebo jiným
6	XR2	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Získání /regenerace rozpouštědel

Obrázek 17. Upravená tabulka Kódování způsobů nakládání s odpady jako zdrojový soubor pro spojování s tabulkami odpadů

Při přiřazování údajů z tabulky Kody_Nakladani do DataFrame Zdrojovy je použita funkce *merge* z modulu Pandas s nastaveným parametrem how = left. Klíčový sloupec má stejný název v obou tabulkách a je nastaven na sloupec ´Kod´. Výsledkem funkce je nová tabulka Zdrojovy_Kody typu DataFrame. Obsahuje sloupce z obou vstupních tabulek a všechny řádky levé tabulky (Zdrojovy) včetně těch, které nemají stejné hodnoty v klíčovém sloupci ´Kod´.

Pro účely spárování dvou DataFrame byla napsána obecná funkce *merge_left()* (Program 6).

```
def merge_left(df1, df2, column1, column2):
    merged_df = pd.merge(df1, df2, left_on=column1,
    right_on=column2, how = 'left')
    return merged_df
```

Program 6. Funkce pro spárování dataframe pomocí merge, how=‘left‘

Pro následnou kontrolu spárování obou DataFrame je vytvořena funkce *kontrola_sparovani*. Vstupní parametry funkcí jsou kontrolovaná DataFrame, a název sloupce, ve kterém se má kontrola provést.

```
def kontrola_sparovani (dataframe,column):
    Nesparovane_radky = dataframe[dataframe[column].isnull()]
    if Nesparovane_radky.empty:
       print(f'Ke vsem radkum leve tabulky byly dohledany
        hodnoty do sloupce {column}')
         return(True, None)
    else:
         Nesparovane_radky_pocet = Nesparovane_radky.shape[0]
         print(f'K {Nesparovane_radky_pocet} radkum nebyla
        dohledana hodnota do sloupce {column}')
         print(Nesparovane_radky)
         return(False,Nesparovane_radky)
```

Program 7. Funkce pro kontrolu spárování dvou DataFrame a výpis nespárovaných řádků

Použití funkce *kontrola_sparovani()* (Program 7) na DataFrame Zdrojovy_Kody bude kontrolovat spárování podle sloupce ´Navyseni_Ubytek´.

Ve zdrojových souborech je množství odpadu uvedeno ve sloupci ´Mnozstvi´ v tunách a v kladných číslech pro všechny operace, tj. pro příjmové i pro výdajové. Pro kontrolu, zda je roční bilance dané ZÚJ rovna 0, je číslo ve sloupci ´Mnozstvi´ násobeno hodnotou -1 nebo +1. Tento výpočet je vložen jako nový sloupec ´Odpad_vKg´ do DataFrame Evident_Kody. Vznikne nová DataFrame Evident_Kody_Mnozstvi. Poté se v rámci ZÚJ provede součet ve sloupci ´Odpad_vKg´ a součet by měl vyjít 0.

Při výpočtu se vyskytuje problém se součty desetinných čísel. Ne u všech ZÚJ vychází součet nulový, byť rozdíl tvoří mnohdy méně než tisícinu tuny (Obrázek 18). Důvodem je, že do ročního hlášení byly zadány hodnoty s velkým množstvím desetinných míst. To nastává například tak, že několik obcí společně odpad vyveze a množství odpadu si pak obce dělí mezi sebou.

	A	E	H	J	N	T	U	V	Y	Z	AA
1	ZmenaMnozstvi	Evident_ZUJ_Nazev_A	Evident_Kraj_Nazev	Evident_TypSubjektu	Partner_ZUJ_Nazev_A	Kod	Indikator	Mnozstvi	Operator	Uvedeni_	Navyseni_Ubytek
7926	713,33	HIVFOS	Středočeský kraj	4	HIVFOS	A00	Produkce	0.71333	(+)	NE	1
9069	-713,333	HIVFOS	Středočeský kraj	4	HESLIP	Předání	Předání	0.713333	(-)	ANO	-1
10991	-0,003										

Obrázek 18. Kontrola bilance ročního zúčtování nevychází nulová

Problém je vyřešen jednak změnou jednotek, tuny se převádí na kilogramy (Program 8). A pak také stanovením hranice, kdy je a kdy není rozdíl přijatelný. Pokud je rozdíl v součtech menší, než jeden kilogram je v kontrolní funkci rozdíl vyhodnocen jako přijatelný.

```
def vlozit_sloupec_prepocet_mnozstvi(df):
    df.insert(loc=0,column='Odpad_vKg',value=(df['Mnozstvi'] *
    df['Navyseni_Ubytek']) * 1000)
    return df
```

Program 8. Funkce pro vložení nového sloupce s přepočtem množství odpadu

Pro kontrolu nulové bilance se nejprve seskupí DataFrame Zdrojovy_Kody_Mnozstvi podle sloupců ´Druh_Odpadu´, ´Evident_ZUJ_Cislo´a ´Evident_TypSubjektu´. ´Odpad_vKg´ je název sloupce, ve kterém se bude provádět součet.

Pro účely seskupování dat byla vytvořena obecná funkce *seskupeni_dat_po_sloupcich* (Program 9). Tato funkce pracuje s metodou knihovny Pandas *groupby()*. Parametry funkce jsou:

- data – DataFrame s daty k seskupení,

- func_column – název sloupce, nad kterým se bude počítat součet pomocí metody *sum()*,
- *group_coulumns – seznam názvů sloupců v tabulce, podle kterých se data seskupí. Jelikož je použita syntaxe *args, je možné předat funkci různý počet argumentů.

Výsledná tabulka se v závěru funkce setřídí sestupně a bude vrácena na výstupu.

```
def seskupeni_dat_po_sloupcich(data, func_column,*group_columns ):
    grouped_data = data.groupby(list(group_columns))[func_column]
    .sum().reset_index()
    sorted_data = grouped_data.sort_values(by=func_column,
    ascending=False)
    return sorted_data
```

Program 9. Funkce pro seskupení dat dle různého počtu sloupců

7.3 Uživatelský výběr podmnožiny dat

7.3.1 Výběrová menu

Cílem je vytvořit dynamická výběrová menu, která by zobrazovala pouze ty hodnoty, jež je možné zvolit s ohledem na data, která jsou aktuálně do modelu načtena. Tento požadavek je programově realizován přes tvorbu unikátních seznamů z jednotlivých sloupců DataFrame Zdrojovy. Pro tyto účely je připravena funkce *unique_list()*. Funkce vyžaduje zadání tří parametrů: df, column_name a start. Parametry jsou popsány v tabulce níže (Tabulka 1).

Tabulka 1. Popis funkce *unique_list()*

Funkce:	**unique_list(df, column_name,** start**)**
Popis funkce:	Funkce vytváří seznam unikátních hodnot z určeného sloupce zadané Data Frame. Na začátek seznamu je vloženo prázdné pole v podobě stringu ´´. Na druhé místo v seznamu je dle volby v parametru start vložen string ´-all-´. Seznam je seřazen dle abecedy.
df	DataFrame
column_name	Jméno sloupce, ve kterém se má vytvořit unikátní seznam hodnot
start	Nepovinný parametr. Pokud je zadán string ´-all-´, vloží ´-all-´ na začátek seznamu.

Programový kód, který výběrová menu vytváří je uveden v Program 10. Do těchto seznamů je vložen kromě prázdného pole, také string ´-all-´, který umožňuje uživateli zadat výběr všech položek v rámci seznamu.

```
def unique_list(df,column_name,start=None):
    u_list = list(df[column_name].unique())
    u_list = sorted(u_list) # seřazení seznamu podle abecedy
    if start == "-all-":
    u_list.insert(0,'-all-') # přidá novou položku "-all-" na po-
    zici 0
    u_list.insert(0,'') # přidá novou položku "" na pozici 0
    return u_list
```

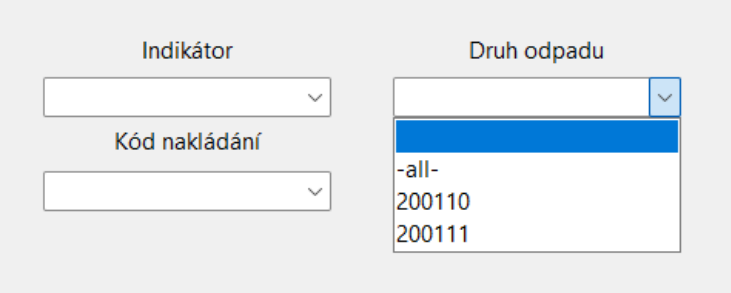
Program 10. Funkce *unique_list()* pro tvorbu dynamických výběrových menu

Funkce je volána pro tvorbu všech výběrových menu na DataFrame Zdrojovy. Příklad volání funkce *unique_list()* je uveden v Program 11.

```
u_list_indikator = unique_list(Zdrojovy,'Indikator','-all-')
u_list_kod = unique_list(Zdrojovy,'Kod','')
u_list_druhOdpadu = unique_list(Zdrojovy, 'Druh_Odpadu','-all-')
u_list_rok = unique_list(Zdrojovy,'Rok','-all-')
```

Program 11. Volání funkce *unique_list()*

Ukázka seznamu vytvořeného pomocí funkce *unique_list()* v GUI na výběrové pole Druh odpadu je uvedena na obrázku (Obrázek 19).

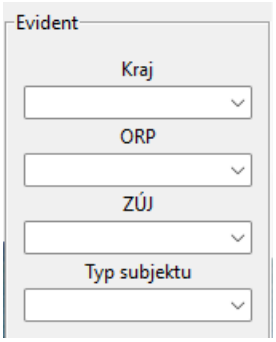


Obrázek 19. Ukázka seznamu vytvořeného pomocí funkce *unique_list()* v GUI

7.3.2 Funkce pro výběr subjektu

Klíčovou fází před aplikací statistických funkcí, grafů, map aj. je výběr dat, nad kterými se budou tyto výstupy realizovat.

Zásadní úlohu hrají výběrové funkce, které na základě volby uživatele vyberou ve vstupní DataFrame řádky, které budou splňovat zadaná kritéria. Výběrová menu pro konkretizaci subjektu (Obrázek 20) obsahují rozevírací seznamy pro volbu kraje, ORP, ZÚJ a typu subjektu. Uživatel může vybrat více položek z jednoho seznamu, a i položky z různých seznamů současně. Pro výběr evidenta a partnera je připravena funkce *vyber_subjektu()*. Popis funkce je uveden v následující tabulce (Tabulka 2).



Obrázek 20. Výběrové menu evidenta v GUI

Tabulka 2. Popis funkce *vyber_subjektu()*

Funkce:	**vyber_subjektu(df, column1, volby1, column2, volby2, column3, volby3, column4, volby4)**
Popis funkce:	Funkce slouží pro výběr řádků datasetu, které splňují podmínky výběru uživatele. Používá se pro výběr evidenta a partnera. Výsledkem funkce je DataFrame s názvem výsledek a je podmnožinou vstupní DataFrame df.
df	Vstupní DataFrame.
column1 až 4	Název sloupce, ve kterém se bude provádět výběr.
volby1 až 4	Seznam voleb uživatele

Pro výběr evidenta nebo partnera je ve funkci *vyber_subjektu()* (Program 12) aplikovaná následující logika. Pokud nejsou uživatelem vybrány konkrétní ORP nebo ZÚJ, tak je výběr dat proveden na úrovni krajů. Pokud uživatel u krajů nevybral nic nebo zvolil ´-all-´ bude do výsledné DataFrame zahrnut celý seznam unikátních hodnot vrácený funkcí

unique_list(). Pokud uživatel vybral konkrétní kraje, vrátí se data pouze pro tyto kraje. Logika je promítnuta do úpravy parametru volby1.

Podobně se řeší výběr typu subjektu. Pokud uživatel nezvolí konkrétní volbu typu subjektu, bude vrácena DataFrame bez tohoto filtru čili bude zahrnovat všechny typy.

Výsledná DataFrame je podmnožinou vstupní DataFrame. Filtrování je realizováno pomocí metody *.isin(),* která testuje, zda prvky v daném sloupci DataFrame jsou obsaženy v daném seznamu hodnot.

```
def vyber_subjektu(df, column1, volby1, column2, volby2, column3,
volby3, column4, volby4) :
    if not (volby2 or volby3):
      if (volby1 == [] or volby1 == ['-all-']):
          volby1 = unique_list(Zdrojovy,column1,'-all-')
          del volby1[0:2]
      else: volby1

    if (volby4 == [] or volby4 == ['-all-']):
        volby4 = unique_list(Zdrojovy,column4,'-all-')
        del volby4[0:2]
    else: volby4
    vysledek = df[((df[column1].isin(volby1)) | (df[column2]
   .isin(volby2)) | (df[column3].isin(volby3)))&(df[column4]
   .isin(volby4))]
    return vysledek
```

Program 12. Funkce *vyber_subjektu()*

Konkrétní volání funkce (Program 13) při výběru evidenta hledá DataFrame, která splňuje podmínky: (hodnota ve sloupci ´Evident_Kraj_Nazev´ je v seznamu volby_evident_kraj **NEBO** hodnota ve sloupci ´Evident_ORP_Nazev´ je v seznamu volby_evident_ORP **NEBO** hodnota ve sloupci ´Evident_ZUJ_Nazev´ je v seznamu voby_evident_nazev) **A současně** je ve sloupci ´Evident_TypSubjektu´ v seznamu volby_evident_typ. Vstupní DataFrame je Zdrojovy_Kody_Mnozstvi.

```
vysledek_evident =
hn.vyber_subjektu(hn.Zdrojovy_Kody_Mnozstvi,'Evident_Kraj_Nazev',
voby_evident_kraj,'Evident_ORP_Nazev',volby_evident_ORP,
'Evident_ZUJ_Nazev',volby_evident_nazev,'Evident_TypSubjektu',
volby_evident_typ)
```

Program 13. Volání funkce *vyber_subjektu()*

Výše bylo uvedeno volání funkce pro výběr evidenta. Obdobně je prováděn výběr partnera.

7.3.3 Funkce pro výběr kritérií

Pro zúžení výběru dat před aplikací statistických funkcí grafů či map je potřeba zohlednit i další kritéria jako jsou: Indikátor, Kód nakládání, Druh odpadu a Rok. Pro tyto účely byla sestavena funkce *vyber_kriterii()* (Tabulka 3).

Tabulka 3. Popis funkce *vyber_kriterii()*

Funkce:	**vyber_kriterii(df, column1, volby1, column2, volby2, column3, volby3, column4, volby4)**
Popis funkce:	Funkce slouží pro výběr řádků datasetu, které splňují podmínky výběru uživatele. Používá se pro výběr kritérií Indikátor, Kód nakládání, Druh odpadu a Rok. Výsledkem funkce je DataFrame s názvem výsledek a je podmnožinou vstupní DataFrame df.
df	Vstupní DataFrame.
column1 až 4	Název sloupce, ve kterém se bude provádět výběr.
volby1 až 4	Seznam voleb uživatele.

Funkce (Program 14) vybírá řádky ze vstupní DataFrame tak, že pokud není uživatelem zvolen konkrétní kód nakládání, tak se výběr bude řídit seznamem pro volbu indikátoru. U roku, druhu odpadu a indikátoru jsou vraceny upravené seznamy voleb. V případě, že uživatel nechal volbu nevyplněnou nebo zvolil ´-all-´ načte se do voleb seznam se všemi dostupnými možnostmi generovaný prostřednictvím funkce *unique_list()*.

```
def vyber_kriterii(df, column1, volby1, column2, volby2, column3,
volby3, column4, volby4) :
    if not (volby2):
      if (volby1 == [] or volby1 == ['-all-']):
          volby1 = unique_list(Zdrojovy,column1,'-all-')
          del volby1[0:2]
      else: volby1
    if (volby3 == [] or volby3 == ['-all-']):
        volby3 = unique_list(Zdrojovy,column3,'-all-')
        del volby3[0:2]
    else: volby3
```

```
    if (volby4 == [] or volby4 == ['-all-']):
        volby4 = unique_list(Zdrojovy,column4,'-all-')
        del volby4[0:2]
    else: volby4
    vysledek = df[((df[column1].isin(volby1)) | (df[column2]
.isin(volby2))) & (df[column3].isin(volby3)) & (df[column4]
.isin(volby4))]
    return vysledek
```

Program 14. Funkce *vyber_kriterii()*

Konkrétní volání funkce (Program 15) při výběru kritérií hledá ty řádky, které splňují podmínky: (hodnota ve sloupci ´Indikator´ je v seznamu volby_indikator **NEBO** hodnota ve sloupci ´Kod´ je v seznamu volby_kod) **A současně** (hodnota ve sloupci ´Druh_Odpadu´ je v seznamu druh_odpadu **A současně** je hodnota ve sloupci ´Rok´ v seznamu volby_rok).

```
vysledek = hn.vyber_kriterii(vysledek_evidentApartner,'Indikator',
volby_indikator,'Kod',volby_kod,'Druh_Odpadu',volby_druhOdpadu,
'Rok',volby_rok)
```

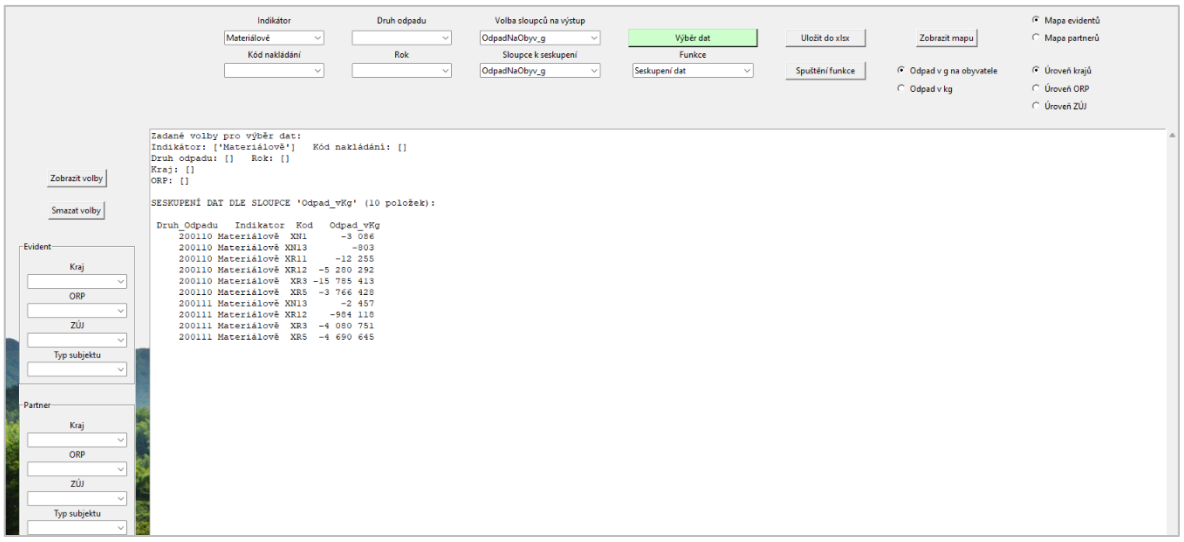
Program 15. Volání funkce *vyber_kriterii()*

7.4 Tvorba GUI

Aby bylo možné ověřit funkčnost navrženého řešení statistického vyhodnocování dat z odpadového hospodářství, vytvořila jsem grafické uživatelské prostředí (GUI). Cílem je, aby se v GUI dal provádět výběr dat ze zdrojových dat na základě voleb uživatele. Nad těmito vybranými daty spouštět připravené funkce pro statistické vyhodnocování, vytvářet grafy a mapy. Výběry dat a výsledky funkcí ukládat do xlsx.

7.4.1 Popis GUI

GUI je vytvořeno pomocí knihovny Tkinter v jazyce Python. Okno aplikace je rozděleno do tří hlavních částí (left_frame, right_frame, text_widget) (Obrázek 21).



Obrázek 21. Obrazovka GUI

Levá část slouží pro:

- Zadání podmínek výběru Evidenta a Partnera prostřednictvím výběrových menu.
- Zobrazení zvolených výběrů.
- Smazání uživatelských voleb.

Programově je tato část pojmenovaná jako left_frame a widgety jsou umísťovány metodou *pack()*.

Pravá část obsahuje:

- Výběrová menu pro zvolení dalších kritérií jako jsou: Indikátor, Kód nakládání, Druh odpadu a Rok.
- Výběrové menu pro možnost volby sloupců, které se budou zobrazovat na výstupu.
- Výběrové menu pro definování sloupců, podle kterých se mají data seskupovat.
- Tlačítko, kterým se spustí funkce pro generování DataFrame s vybranými daty.
- Výběrové menu pro zvolení výpočtové funkce.
- Tlačítko pro spuštění vybrané výpočtové funkce.
- Tlačítko pro uložení filtrovaných nebo vypočítaných dat do souboru s příponou xlsx.
- Přepínací tlačítka pro specifikaci mapy. Zda má být mapa zpracována z pohledu evidentů či partnerů, zvolení hierarchické úrovně (kraj, ORP, ZÚJ) a volba typu zobrazovaných dat (Odpad v g na obyvatele, Odpad v kg).
- Tlačítko pro zobrazení mapy.

Widgety v pravé části jsou tabulkově uspořádány pomocí metody *grid()*. Pravá část je v programu pojmenována right_frame.

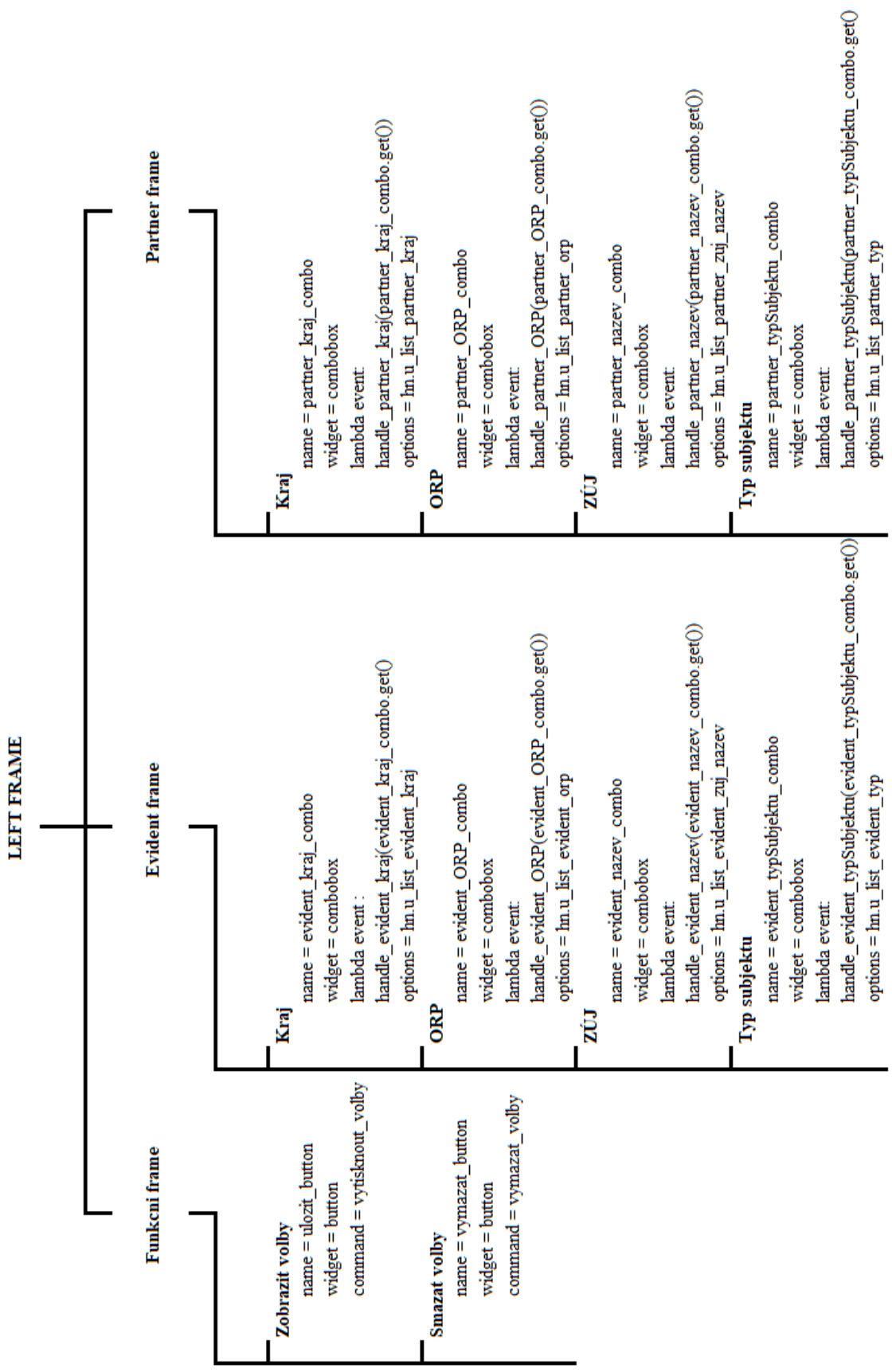
Část okna aplikace, která slouží pro zobrazování výstupů je v programu pojmenována jako text_widget a v okně aplikace zabírá největší prostor. Pro možnost zobrazení rozsáhlých výstupů je text_widget opatřen svislým posuvníkem.

7.4.2 Programový koncept tvorby GUI

Hlavní okno aplikace je vytvořeno pomocí třídy Tk(). Jednotlivé prvky uživatelského rozhraní jsou umístěny do hlavního okna aplikace pomocí správců geometrie. Left_frame pomocí pack*()* a right_frame využívá tabulkové uspořádání prostřednictvím *grid()*. Hlavní smyčka aplikace je spouštěna pomocí metody *mainloop()*. Tím je zajištěno zobrazení GUI a jeho interaktivnost.

7.4.2.1 Left_frame struktura a popis funkčnosti widgetů

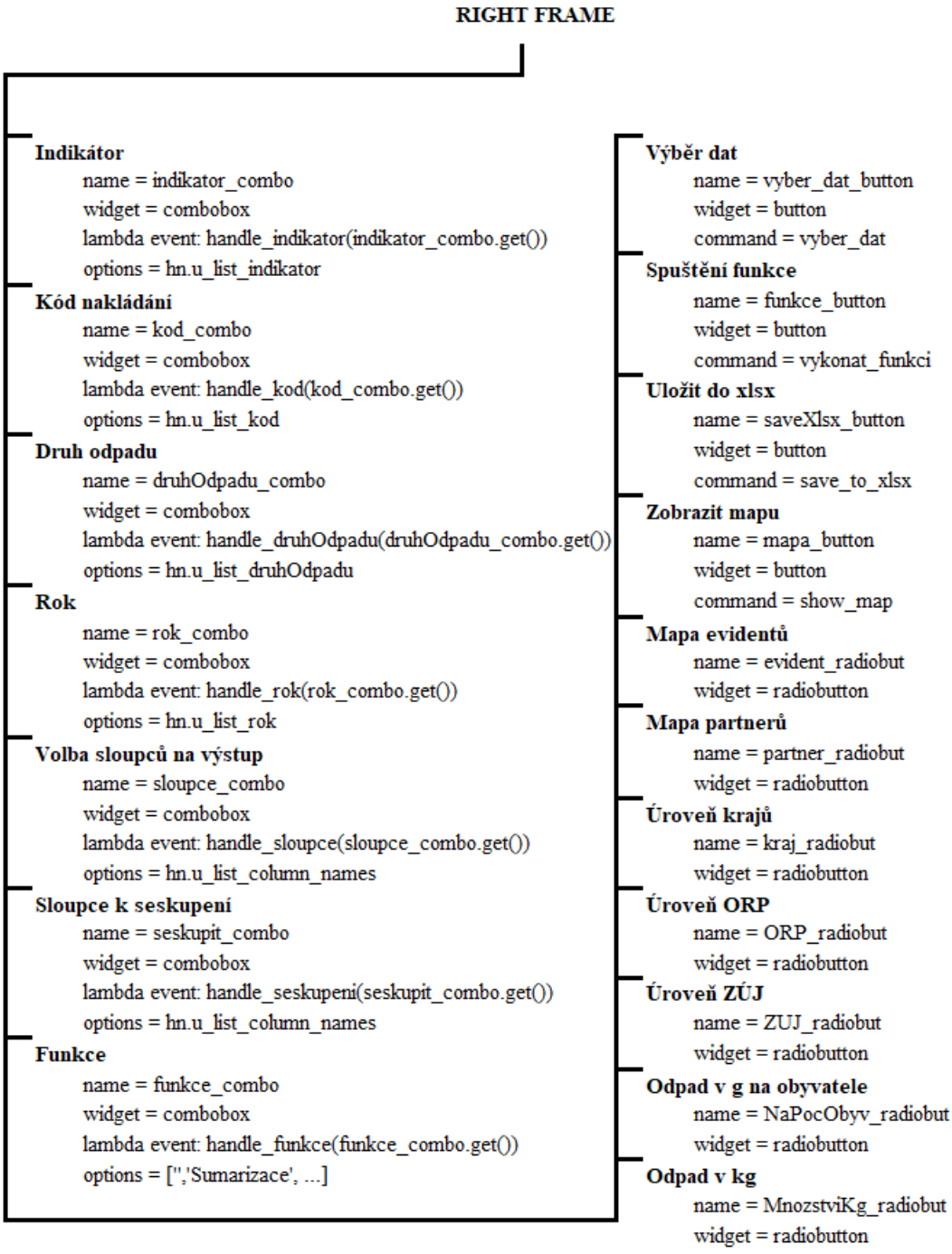
V Left_frame (Obrázek 22) jsou widgety umístěny podle přirozeného toku layoutu směrem od shora dolů. Výběrová menu pro specifikaci evidenta a partnera jsou zabalena do evident_frame a partner_frame. Tlačítka pro zobrazení a smazání voleb jsou umístěna v rámci funkcni_frame.



Obrázek 22. Přehled základních prvků v GUI Left frame

7.4.2.2 Right_frame struktura a popis funkčnosti widgetů

Widgety v Right_frame jsou uspořádány tabulkově pomocí funkce *grid()*. Každému widgetu je nastavena souřadnice v tabulce: číslo řádku a číslo sloupce podle které, je umístěn v kontejneru.



Obrázek 23. Přehled základních prvků GUI v Right frame

7.4.3 Výběrová menu – Combobox

Jedním z nejčastěji použitým widgetem v rámci GUI jsou výběrová menu. Programově jsou vytvářena podobně, proto logiku vysvětlena jen na jednom např. výběrové menu pro výběr kraje u evidenta (Program 16).

```
evident_kraj_label = tk.Label(evident_frame, text="Kraj")
evident_kraj_label.pack(side=tk.TOP)
options = hn.u_list_evident_kraj
evident_kraj_combo=ttk.Combobox(evident_frame, value=options)
evident_kraj_combo.bind("<<ComboboxSelected>>", lambda event: han-
dle_evident_kraj(evident_kraj_combo.get()))
evident_kraj_combo.current(0)
evident_kraj_combo.pack(side=tk.TOP)
```

Program 16. Kombinovaný seznam – Combobox pro výběr kraje u evidenta

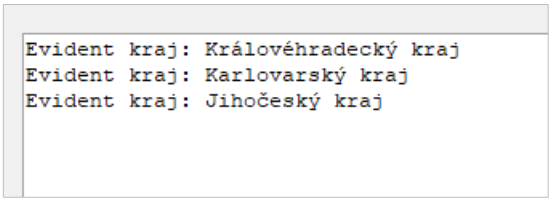
Výběrová menu jsou tvořena kombinovaným seznamem tzv. Combobox. V rámci comboboxu je definovaná proměnná options, která obsahuje seznam možností k výběru uživatelem. Tento seznam je vytvářen pomocí funkce *unique_list()*. Funkce vybere unikátní hodnoty z patřičného sloupce ze zdrojových dat (zdrojové DataFrame), přidá k nim prázdné pole (string ´ ´) a v případě potřeby možnost ´-all-´.

Hodnoty comboboxu jsou nastaveny na seznam proměnné options. V případě, že uživatel vybere některou z možností v comboboxu, spustí se událost ComboSelected, ke které je pomocí metody *bind()* přiřazena funkce *handle_evident_kraj()* (Program 17).

```
def handle_evident_kraj(selection):
    volby_evident_kraj.append(selection)
    text_widget.insert('1.0', f"Evident kraj: {selection}\n")
```

Program 17. Funkce *handle_evident_kraj()*

Funkce *handle_evident_kraj(selection)* má vstupní parametr selection. Funkce umožňuje vícenásobný výběr a přidává hodnotu uloženou v parametru selection do seznamu volby_evident_kraj. Zároveň doplňuje text „Evident kraj: “ o volbu z parametru selection. Tento text se zobrazuje v text widgetu (Obrázek 24).



Obrázek 24. Zobrazení volby uživatele v text widgetu

Seznam volby_evident_kraj je definován na začátku programu a bude používán jako vstup do funkce *vyber_evident_partner_kriteria()*, stejně jako ostatní volby uživatele, které jsou provedeny přes combobox.

Výchozí hodnota comboboxu je nastavena na první položku seznamu options čili string ´ ´, který je zobrazen jako prázdné pole.

7.4.4 Tlačítko – Button

Tlačítka jsou v GUI vytvářena velmi podobně (Program 18). Důraz se však klade na to, jakou funkci spouští. Obecně se tlačítku definuje umístění v rámci frame, zobrazovaný text a spouštěná funkce.

```
ulozit_button = tk.Button(funkcni_frame, text="Zobrazit vol-
by",command=vytisknout_volby)
ulozit_button.pack(side=tk.TOP, pady=(20,0))
```

Program 18. Kód pro tvorbu tlačítka s popiskem

7.4.4.1 Tlačítko Zobrazit volby – funkce vytisknout_volby()

Tlačítko Zobrazit volby spouští funkci s názvem *vytisknout_volby()*. Funkce je připravena proto, aby si uživatel mohl zkontrolovat či připomenout, jaké volby pro filtrování vstupních dat zvolil.

Funkce zobrazuje v text widgetu jednotlivé seznamy voleb uživatele (Obrázek 25).

```
ZADANÉ VOLBY:

Vybraný výpočet: ['Sumarizace']

Sloupce na výstup: []

Seskupení podle sloupců: []

Indikátor: ['Produkce']
Kód nakládání: []
Druh odpadu: ['200110']
Rok: ['2021']

Evident kraj: ['Jihomoravský kraj', 'Kraj Vysočina', 'Karlovarský kraj']
Evident ORP: ['Zlín']
Evident ZÚJ: []
Evident typ subjektu: ['1']

Partner kraj: []
Partner ORP: []
Partner ZÚJ: []
Partner typ subjektu: []
```

Obrázek 25. Zobrazení seznamů s volbami uživatele v text widgetu

7.4.4.2 Tlačítko Smazat volby – funkce vymazat_volby()

Tlačítko z left_frame Smazat volby spouští funkci *vymazat_volby()*. Jejím hlavním úkolem je umožnit uživateli smazat své volby a vrátit se zpět s výběrem do výchozího stavu tak, aby mohl provést výběr jiný (Program 19).

```
def vymazat_volby():
    text_widget.delete('1.0','end')
    volby_evident_kraj.clear()
    volby_evident_ORP.clear()
     ......
    evident_kraj_combo.current(0)
    evident_ORP_combo.current(0)
     ......
    funkce_combo.configure(state="disabled")
    saveXlsx_button.configure(state='disabled')
    mapa_button.configure(state='disabled')
    funkce_button.configure(state='disabled')
```

Program 19. Ukázka zkrácené funkce *vymazat_volby()*

Funkce jako první krok smaže všechny textové hodnoty v text widgetu pomocí metody *delete()*. Poté pomocí metody *clear()* vymaže hodnoty v seznamech, kam se ukládají volby uživatele. Dále skrz metodu *current(0)* nastaví výběrové comboboxy na první položku výběrového seznamu tj. prázdné pole. Pro widgety Výběr funkce, Spuštění funkce, Uložit do

xlsx a Zobrazit mapu nastavuje stav na „disabled“. Tím se tlačítko deaktivuje a znemožňuje uživateli na něj kliknout či vybírat z nabídky.

7.4.4.3 Tlačítko Výběr dat – funkce vyber_dat()

Tlačítko s názvem Výběr dat spouští funkci *vyber_dat()*, která sama o sobě spouští funkci *aktivovat_vyber()* a *vyber_evident_partner_kriteria()*. Snahou bylo, aby uživatel musel nejprve vybrat data a následně až potom mohl volit výpočtové funkce či tvořit mapu nad těmito filtrovanými daty.

Funkce *aktivovat_vyber()* (Program 20) nastavuje proměnnou vyber_dat_stisknuto na hodnotu True, což signalizuje, že uživatel vybral data. Následně jsou widgety Volba funkce, Spuštění funkce, Uložit do xlsx a Zobrazit mapu změnou stavu aktivovány pro použití uživatelem.

```
def aktivovat_vyber():
    global vyber_dat_stisknuto
    vyber_dat_stisknuto = True
    funkce_combo.configure(state='readonly')
    saveXlsx_button.configure(state='normal')
    mapa_button.configure(state='normal')
    funkce_button.configure(state='normal')
```

Program 20. Funkce *aktivovat_vyber()*

Funkce *vyber_evident_partner_kriteria()* je zásadní pro výběr dat. Jejím hlavním úkolem je vyfiltrovat data dle voleb uživatele a výsledky zobrazit v textovém widgetu. Z důvodu délky funkce je její celé znění umístěno v příloze (P VII).

Funkce nejprve načte globální proměnné výsledek_excel a vyber_dat_vysledek. Tyto proměnné slouží pro uchování výsledků a jejich pozdější použití ve výpočtových funkcích.

Dále je použita funkce *vyber_subjektu()* na DataFrame se zdrojovými daty Zdrojovy_Kody_Mnozstvi. Parametry funkce jsou seznamy s uloženými volbami z comboboxu v evident_frame. Výsledný dataset je uložen do DataFrame výsledek_evident.

DataFrame výsledek_evident je použita jako vstupní parametr pro funkci *vyber_subjektu()* opět pro užší výběr dat. Tentokrát jsou jako parametry funkce použity seznamy s uloženými volbami pro partnera z comboboxu v partner_frame. Výsledkem je DataFrame s názvem výsledek_evidentApartner.

DataFrame výsledek_evidentApartner je opět vstupem do funkce *vyber_kriterii()*. Jako parametry funkce jsou opět použity seznamy s uloženými volbami indikátoru, kódu nakládání, druhu odpadu a roku. Vzniklá DataFrame se jmenuje vysledek.

Globální proměnné vysledek_excel a vyber_dat_vysledek jsou nastaveny na DataFrame vysledek.

Vyfiltrovaná data se zobrazují do text widgetu a uživatel má možnost si zvolit, jaké sloupce ze zdrojové DataFrame chce zobrazit. Pokud toho chce uživatel využít, použije se pro zobrazení právě seznam voleb z comboboxu Volba sloupců na výstup (Obrázek 26). Pokud toho nechce využít, použije se již předem definovaný seznam sloupců volby_sloupce_univ.

```
VÝPIS FILTROVANÝCH DAT DLE VÝBĚRU EVIDENTA, PARTNERA A KRITÉRIÍ (10 položek):

   Evident_Kraj_Nazev Evident_ZUJ_Nazev_A Indikator  Kod OdpadNaObyv_g Druh_Odpadu  Rok Odpad_vKg Pocet_Obyvatel
    Středočeský kraj              HOSKIF Spalování XD10           -30      200110 2021      -964         31 807
    Středočeský kraj              HUPDEL Spalování XD10        -3 494      200110 2021   -63 390         18 138
      Jihočeský kraj              KESHIN Spalování XD10            -4      200110 2021       -95         22 300
        Ústecký kraj              KIBRIB Spalování XD10        -1 179      200110 2021    -3 800          3 221
       Plzeňský kraj              KINFEB Spalování XD10            -6      200110 2021    -1 081        168 265
Moravskoslezský kraj              KISFIB Spalování XD10             0      200110 2021      -256        282 265
      Liberecký kraj              KATFAM Spalování XD10            -1      200110 2021       -85         44 777
        Zlínský kraj              RUFNOL Spalování XD10            -2      200110 2021      -170         73 389
      Olomoucký kraj              REPDEP Spalování XD10           -10      200110 2021      -462         43 344
       Kraj Vysočina              DABHIT Spalování XD10            -1      200110 2021       -15          9 851
```

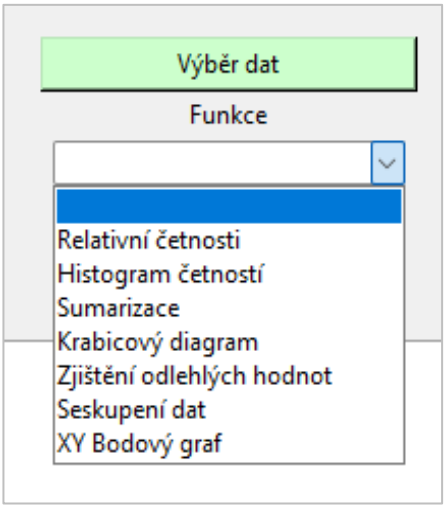
Obrázek 26. Příklad výstupu z funkce *vyber_evident_partner_kriteria()* s uživatelem volenými sloupci na výstup

V závěru funkce se ověřuje, zda DataFrame obsahuje data. Pokud ano, tak se upravuje formátování sloupců ´Odpad_vKg´, ´Pocet_Obyvatel´ a ´OdpadNaObyv_g´. Formátování je realizováno funkcí *format_column(df)*. Funkce přijímá DataFrame a aplikuje formát zobrazení čísel s oddělením tisíců na konkrétní sloupce.

V případě, že výsledná DataFrame neobsahuje data, je vrácena do text widgetu informativní hláška.

7.4.4.4 Combobox Funkce

Poté, co jsou data vybrána a zobrazena v text widgetu, je aktivován combobox nazvaný Funkce (Obrázek 27). Slouží pro uživatelský výběr výpočtové funkce. V options se nachází seznam možností, které může uživatel v kombinovaném widgetu vybrat. Do seznamu options jsou zadány názvy funkcí, které jsou uživateli dostupné pro zpracování dat. První položka v seznamu je prázdný řetězec, který je výchozím stavem widgetu a značí, že žádná funkce nebyla vybrána.



Obrázek 27. Combobox pro výběr výpočtové funkce

7.4.4.5 Tlačítko Spuštění funkce – funkce vykonat_funkci

Po zvolení výpočtové funkce uživatel stiskne tlačítko pojmenované Spuštění funkce a zavolá se funkce *vykonat_funkci()* (Program 21). Funkce nejprve načte hodnotu vybrané funkce z comboboxu a uloží ji do proměnné vybrana_funkce. Vymaže veškerý obsah text widgetu a vloží do něj text, který informuje uživatele o tom, jakou zvolil výpočtovou funkci. Ověří se, že je zvolená funkce ve slovníku funkce_dict. Pokud ano, funkci spustí. Pokud ne, vypíše informativní zprávu do textového widgetu.

```
def vykonat_funkci():
    vybrana_funkce=funkce_combo.get()
    text_widget.delete('1.0','end')
    text_widget.insert("end",f"Vybrána funkce: {vybra-
    na_funkce.upper()}\n")
    if vybrana_funkce in funkce_dict:
        funkce_dict[vybrana_funkce]()
    else:
        text_widget.insert("end","Nejprve vyberte data a poté vy-
        berte funkci ze seznamu.")
```

Program 21. Funkce *vykonat_funkci()*

7.4.4.6 Tlačítko Uložit do xlsx – funkce save_to_xlsx()

Tlačítko Uložit do xlsx použije uživatel v případě, že potřebuje filtrovanou DataFrame nebo výsledky výpočtových funkcí uložit do souboru formátu xlsx. Filtrovaná DataFrame

se ukládá včetně všech 36 sloupců, které obsahuje zdrojová DataFrame Zdrojovy_Kody_Mnozstvi. V případě stisknutí tlačítka až poté, co byla použita výpočtová funkce, uloží se výstup výpočtové funkce, který je zobrazený v textovém widgetu.

Stiskem tlačítka Uložit do xlsx se zavolá funkce *save_to_xlsx()* (Program 22). Ta nejprve testuje, zda globální proměnná vysledek_excel obsahuje nějaká data. Pokud je proměnná nenulová, funkce vytvoří dialogové okno, kde může uživatel vybrat umístění a název souboru pro uložení dat. Funkce uloží data, které jsou v proměnné vysledek_excel a zobrazí informační dialogové okno s hlášením, že data byla úspěšně uložena. Pokud je proměnná vysledek_excel rovna nule (tj. neobsahuje žádná data), funkce zobrazí upozornění, že nebyla nalezena žádná data k uložení.

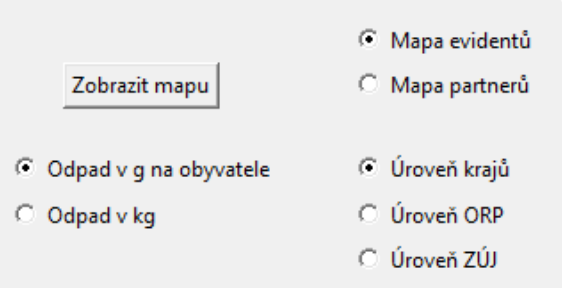
```
def save_to_xlsx():
    if vysledek_excel is not None:
           file_name = filedialog.asksaveasfilename (defaultexten-
           sion='.xlsx')
        if file_name:
           vysledek_excel.to_excel(file_name, index=True, hea-
           der=True)
           messagebox.showinfo("Uloženo", "Data byla uložena do
           Excelu.")
    else:
           messagebox.showwarning("Chyba", "Nebyla nalezena žádná
           data k uložení.")
```

Program 22. Funkce *save_to_xlsx()*

Do souboru je uložena DataFrame včetně názvů sloupečků a indexů řádků. Indexy jsou vloženy do sloupce A.

7.4.5 Přepínací tlačítka pro specifikaci tvorby mapy

Při tvorbě mapy je umožněno uživateli určit, zda chce vidět data z pohledu evidenta nebo partnera. Může si zvolit, zda hodnoty v mapě zobrazit v gramech na obyvatele nebo množství odpadu na danou územní jednotku v kg. A rovněž si může zvolit úroveň mapového podkladu (kraje, ORP, ZÚJ). Tyto volby jsou realizovány skrze přepínací tlačítka (Obrázek 28).



Obrázek 28. Přepínací tlačítka pro specifikaci tvorby mapy

Přepínací tlačítka neboli Radiobuttons jsou typem widgetu, který umožňuje uživateli vybrat pouze jednu možnost ze skupiny navzájem se vylučujících možností. Každý radiobutton má společnou proměnnou, která určuje, který radiobutton je aktuálně vybrán.

Přepínací tlačítka jsou v aplikaci řešena obdobně, proto uvedu příklad pro skupinu radiobuttons zaštiťující výběr subjektu, z jehož pohledu jsou data v mapě zobrazena (Program 23).

```
# Proměnné pro radiobutton
subjekt_radiobut_value = tkinter.StringVar()
subjekt_radiobut_value.set('1')
# Vytvoření radiobuttons
evident_radiobut = tkinter.Radiobutton(right_frame, text="Mapa
evidentů", variable=subjekt_radiobut_value, value="1")
evident_radiobut.grid(row=3, column=6, padx=20, pady=0)
partner_radiobut = tkinter.Radiobutton(right_frame, text="Mapa
partnerů", variable=subjekt_radiobut_value, value="2")
partner_radiobut.grid(row=4, column=6, padx=20, pady=0)
```

Program 23. Radiobuttons pro výběr subjektu z jehož pohledu bude mapa zobrazena

Nejprve se vytvoří proměnná subjekt_radiobut_value, která bude sloužit pro ukládání hodnoty vybraného radiobuttonu. Výchozí hodnota proměnné je nastavena pomocí metody set() na 1. Poté jsou vytvořeny dva radiobuttony. Prvním je „Mapa evidentů“ s přidělenou hodnotou value = 1. Druhým je „Mapa partnerů“ s hodnotou value = 2. Oba radiobuttony tedy sdílejí stejnou proměnnou, ale mají různé hodnoty. S touto hodnotou bude pracovat funkce *show_map()* až se bude mapa vytvářet.

7.4.6 Tlačítko Zobrazit mapu – funkce show_map()

Poté, co uživatel specifikuje, jakým způsobem chce nechat zobrazit mapu, stiskne tlačítko Zobrazit mapu, které zavolá funkci *show_map()*. Funkce je popsána v kapitole 7.5.1. Cílem

funkce *show_map()* je vykreslit mapu České republiky a dle uživatelem zvolené geografické úrovně zobrazit hodnoty v barevné škále. Současně s mapou se do text widgetu zobrazí data, která jsou v mapě vizualizována. Mapa se otevírá v novém okně pomocí knihovny Matplotlib.

7.5 Statistické zpracování dat

Poté, co je připraveno grafické uživatelské prostředí, které umožňuje uživateli získat výběrový soubor, volit výpočtové funkce a vizualizovat výsledky, můžeme přistoupit k samotné tvorbě těchto funkcionalit, které povedou ke statistickému zhodnocení dat.

7.5.1 Zobrazení dat v mapě

Jedním z velmi mocných nástrojů datové analýzy je mapa. Mapa dokáže vizualizovat geografické rozložení dat v různých regionech a umožňuje nám rychle porovnat úroveň daného ukazatele v různých oblastech. Prostřednictvím mapy je možné vizuálně identifikovat vzorce a trendy, které bychom jinak mohli přehlédnout. Z toho důvodu byla připravena funkcionalita zobrazení mapy nad výběrovým souborem dat v podobě funkce *show_map()*.

Plné znění funkce je uvedeno v příloze (P VIII). Pro zobrazení mapy je využito knihovny Matplotlib, která na svém vstupu přijímá mapový podklad s hodnotami, které chceme vizualizovat. Jako mapový podklad byl zvolen geografický datový soubor vektorového formátu GeoJSON, který byl získán na Portálu otevřených dat jako výstup článku autora Michala Škopa [32]. Využívají se tři soubory obce-simple.json, orp-simple.json a kraje-simple.json. Dále je pracováno s Lexikonem obcí, což je soubor typu CSV, který obsahuje seznam všech ZÚJ s uvedením jejich čísla, názvu, příslušné ORP a kraje, a počtu obyvatel [33]. Jako další zdroj byl využit seznam obcí české republiky ve formátu JSON [34] a soubor s počty obyvatel České republiky [35].

7.5.1.1 Popis programového řešení mapy

Na začátku funkce *show_map()* se ověřuje, zda globální proměnná vyber_dat_vysledek je nenulová. Tato proměnná by totiž měla obsahovat vyfiltrovaná vstupní data, která vrací funkce *vyber_evident_partner_kriteria()*. Kdyby nic neobsahovala, program by vrátil chybovou hlášku.

Poté se načítají pomocí funkce *gdp.read_file()* geografické soubory pro úrovně krajů, ORP a ZÚJ. Soubory jsou formátu geoJSON a obsahují geografické informace o regionech České republiky a jsou využívány k sestavení mapy.

Následně se testováním podmínek zjišťuje, které radiobuttony uživatel zvolil při volbě subjektu, geografické úrovně a hodnot (Obrázek 28). Podle jeho voleb se přiřazují proměnným hodnoty, které budou později použity pro načítání souboru s mapovými podklady, pro výpočet dat a pro jejich následné zobrazení. Rovněž budou informace sloužit pro tvorbu titulku mapy.

Pro zvolenou geografickou úroveň se počítá pomocí funkce *odpadNaObyvatele_g()* jaký je odpad na obyvatele daného území (Program 24) a uloží se do DataFrame mapa_data.

```
def odpadNaObyvatele_g(df_filtered,column_grouped,
    df_lexikon,column_lexikon):
    odpad = seskupeni_dat_po_sloupcich(df_filtered,'Odpad_vKg',
    column_grouped)
    odpad['Odpad_vKg'] = odpad['Odpad_vKg'].abs()
    obyvatele = seskupeni_dat_po_sloupcich(df_lexikon,
    'Pocet_Obyvatel',column_lexikon)
    odpad_obyvatele = obyvatele.merge(odpad,left_on=
    column_lexikon, right_on = column_grouped, how='left')
    odpad_obyvatele['Odpad_vKg'] = odpad_obyvatele['Odpad_vKg']
    .fillna(value=0)
    odpadNaObyv_g = vlozit_sloupec_prepocet_odpadNaPocetObyv
    (odpad_obyvatele)
    return odpadNaObyv_g
```

Program 24. Funkce *odpadNaObyvatele_g()*

Funkce vytvoří DataFrame odpad, ve které sečte ´Odpad_vKg´ pro každou unikátní hodnotu ve sloupci ´column_grouped´, to může být např. ´Evident_ORP_Cislo´. Na sečtený odpad se aplikuje funkce *abs()*, která zajistí, že všechny součty budou kladné. Je to z toho důvodu, že některé druhy nakládání jsou uváděny se záporným znaménkem, protože v ročním zúčtování odpad odečítají. Jsou to například indikátory: Předání, Skládkování, Spalování a jiné.

Dále je vytvořena DataFrame obyvatele, do které se vypočítá celkový počet obyvatel pro každou unikátní hodnotu ve sloupci ´column_lexikon´ v Lexikonu obcí, tou může být např. ´ORP_Cislo´.

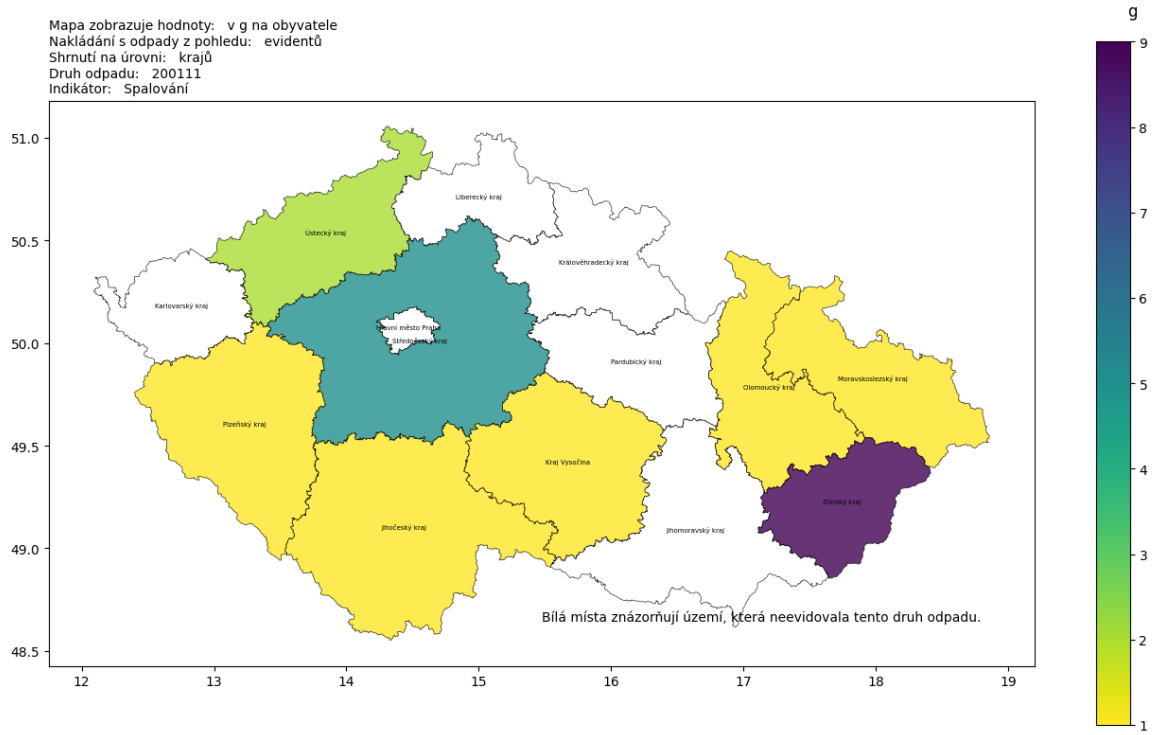
Pomocí funkce *merge()* se propojí DataFrame odpad a DataFrame obyvatele do jedné DataFrame odpad_obyvatele. Tam, kde nedošlo ke spojení se vloží hodnota NaN, kterou je nahrazena nulou 0 pomocí funkce *fillna()*. Jako poslední krok se vloží nový sloupec odpadNaObyv_g, kde se pomocí funkce *vlozit_sloupec_prepocet_odpadNaPocetObyv()* vypočte odpad na obyvatele jako podíl dvou sloupců a hodnota se vynásobí 1000. Tím dojde k převedení hodnot z kilogramů na gramy (Program 25).

```
def vlozit_sloupec_prepocet_odpadNaPocetObyv(df):
    df.insert(loc=0,column='OdpadNaObyv_g',value=(df['Odpad_vKg']
/ df['Pocet_Obyvatel'])*1000)
    return df
```

Program 25. Funkce *vlozit_sloupec_prepocet_odpadNaPocetObyv()*

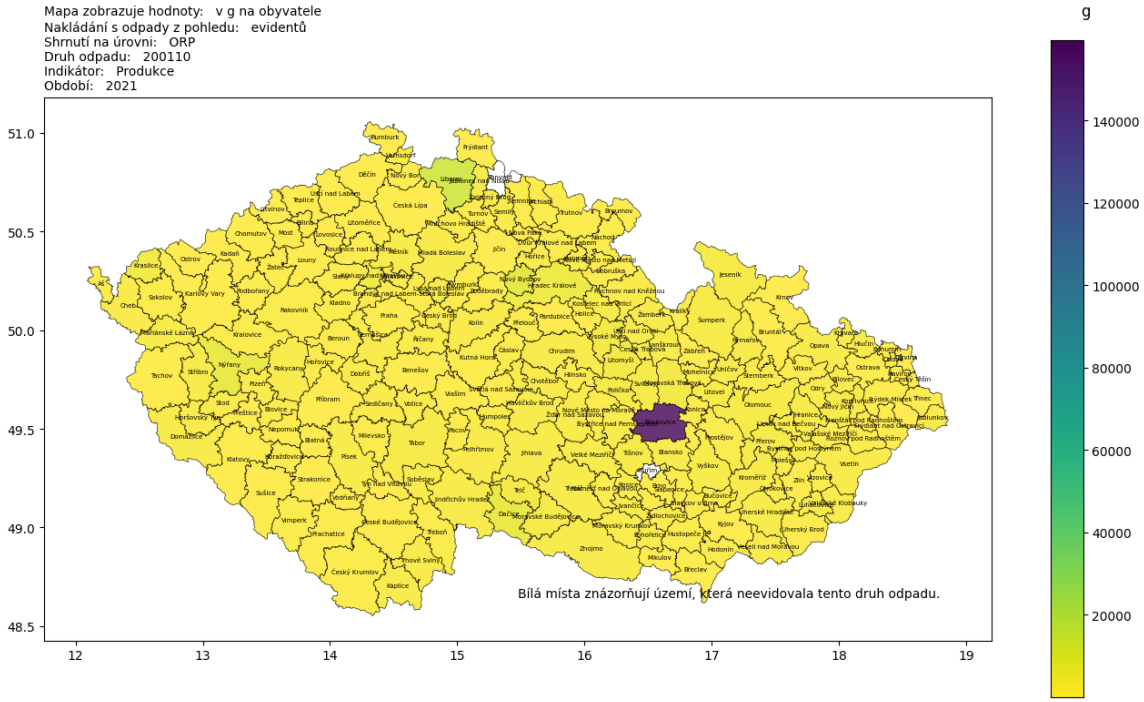
Připravenou DataFrame mapa_data sloučíme s příslušným mapovým podkladem opět pomocí funkce *merge()*. Vznikne DataFrame s názvem gdf_merged. U gdf_merged ještě zaměníme případné NaN hodnoty za 0 a sloupce ´OdpadNaObyv_g´a ´Odpad_vKg´ zaokrouhlíme na celá čísla nahoru.

Zaokrouhlení nahoru je děláno proto, protože je nutné se v mapě vypořádat s oblastmi, které nevykazovaly žádný odpad. Takové oblasti budou v mapě zobrazeny bílou barvou (Obrázek 29) a barevně pak budou vykresleny až oblasti, které vykazují hodnotu odpadu 1 gram a vyšší.



Obrázek 29. Ukázka mapy, ve které jsou oblasti, které nevykazují odpad, zobrazeny bíle

Při zobrazování dat v mapě se vyskytl problém s vybočujícími hodnotami. Pokud ve výběrovém souboru byly oblasti, které svými hodnotami několikanásobně převyšovaly ostatní, způsobily, že mapa byla téměř jednolitá (Obrázek 30). Barevně se odlišovaly pouze vybočující oblasti a u zbytku nebyly drobnější rozdíly viditelné.



Obrázek 30. Ukázka mapy, ve které vybočující hodnoty způsobují jednolitost mapy

Z tohoto důvodu je určena horní hranice pro zobrazení hodnot v rámci legendy a je nastaveno, že hodnoty přesahující danou hranici budou vykresleny černou barvou. Pro tyto účely byla vytvořena proměnná pocet_odlehlych_hodnot, podle které se určuje, zda bude horní hranici škály určovat maximální hodnota z výběrového souboru u sledované veličiny nebo bude tato horní hranice určena výpočtem z funkce *zjisteni_hranic()*.

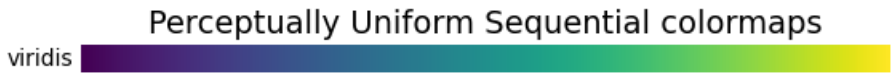
Samotná mapa je vykreslena zejména díky části kódu uvedeného v Program 26.

```
cmap = cm.get_cmap('viridis')
cmap = cmap.reversed()
cmap.set_over('black')
cmap.set_under('white')
# nastavení rozsahu hodnot pro barevnou mapu
vmin = 1
vmax = upper_limit_scale  # nastavíme max hodnotu v
hodnotách jako vrchol legendy
norm = plt.Normalize(vmin=vmin, vmax=vmax)

fig, ax = plt.subplots()

# použití metody plot() pro zobrazení mapy
gdf_merged.plot(column = hodnoty,
                cmap = cmap,
```

```
    ax=ax,
    norm=norm,
    legend = True,
    edgecolor='black',
    linewidth=0.5,
    alpha=0.8,
    )
```

Program 26. Kód pro vytvoření barevné mapy a vykreslení dat na mapu

Nejprve se definuje barevná mapa, ve které se použije barevná mapa pojmenovaná viridis (Obrázek 31). Ta se následně obrátí tak, aby se nejvyšší hodnoty mapovaly na nejnižší barvy. Tzn. nejnižší hodnoty jsou zbarveny žlutě, nejvyšší tmavomodře.



Obrázek 31. Barevná mapa viridis [36]

Dále se pomocí *cmap.set_over()* nastaví černá barva pro hodnoty převyšující vmax a *cmap.set_under()* nastaví bílou barvu pro hodnoty podcházející daný rozsah tj. vmin. Stanoví se hodnoty vmin a vmax, které slouží jako parametry při normalizaci dat. Graf se následně vytvoří pomocí *plt.subplots()* a pomocí metody *.plot()* se data vykreslí na mapu.

Mapa je opatřena v pravém spodním rohu plochy grafu popiskem (Obrázek 32). Který v případě výskytu vybočujících hodnot informuje o této skutečnosti a uvádí hodnotu, od které jsou už oblasti vykresleny černě. Další část popisku informuje o významu bílých oblastí a uvádí se pouze v případě, že tyto oblasti v mapě jsou.

Obrázek 32. Informativní popisky pro přesahující a podcházející hodnoty

Do mapy jsou také přidány názvy jednotlivých oblastí dle geografických úrovní kraje a ORP (Program 27). Na úrovni ZÚJ názvy uvedeny nejsou z důvodu nečitelnosti při vykreslení takového množství dat. Pomocí cyklu for se prochází všechny řádky v GeoDataFrame gdf_merged. Při každém průchodu se použije metoda *plt.annotate()*, která přidá anotaci

s textem příslušného řádku ze sloupce ´NAZEV´ na pozici danou těžištěm row[´geometry´] polygonu.

```
if uzemi_radiobut_value.get() != '3':
   for index, row in gdf_merged.iterrows():
     plt.annotate(text=row['NAZEV'],
             xy=row['geometry'].centroid.coords[0],
             horizontalalignment='center',
             color='black',
             fontsize=5)
```

Program 27. Kód pro zobrazení názvů oblastí v mapě

Mapa ve své horní části obsahuje titulek, který je složen z řádkových informací o kritériích, které uživatel při volbě výběrového souboru zvolil (Obrázek 33). Pokud uživatel nezvolil nějaký parametr, nebude v titulku uveden. Při tvorbě titulku se vychází z hodnot proměnných radiobuttonů a seznamů vytvořených z voleb prostřednictvím comboboxů.

Obrázek 33. Příklad titulku mapy složeného z uživatelských voleb

Seznamy jsou prostřednictvím funkce *create_title_from_list()* poskládány z jednotlivých prvků listu, ke kterým je připojena čárka a mezera. Následně jsou poslední dva znaky odstraněny. Příklad tvorby titulku pro evident_kraj uvádím v Program 28.

```
def create_title_from_list(list):
     title = ""
     for item in list:
          title = title + str(item) + ", "
     title = title[:-2]  # odstranění posledních dvou znaků
     return title

if volby_evident_kraj:
     text = create_title_from_list(volby_evident_kraj)
     evident_kraj = f'Evident kraj:   {text}      '
else: evident_kraj= ''
```

```
# Sestavení titulku mapy podle voleb uživatele
title_text = f'{hodnoty_text}{subjekt_text}{uzemi_text} {evi-
dent_kraj}{evident_ORP}{evident_ZUJ}{evident_typ}{partner_kraj}
{partner_ORP}{partner_ZUJ}{partner_typ}{odpad}{indikator} {nakla-
dani}{obdobi}'
plt.title(title_text,ha='left',loc='left',fontsize=10)
```

Program 28. Příklad tvorby titulku mapy

Text titulku je sestaven z jednotlivých řetězců. Vytvoření nadpisu provede funkce *plt.title()*, ve které je dále specifikováno umístění titulku v rámci obrázku vlevo nahoře.

7.5.1.2 Popis programového řešení výstupu do text widgetu

Současně s tvorbou mapy je v rámci funkce *show_map()* řešeno také zobrazení hodnot, ze kterých je mapa vytvořena do text widgetu (Obrázek 34).

```
Data pro vznik mapy:

 Kraj_Nazev           Kraj_Cislo OdpadNaObyv_g Pocet_Obyvatel Odpad_vKg
       Kraj Vysočina CZ063             1          504 025          297
Královéhradecký kraj CZ052             2          542 583          803
    Středočeský kraj CZ020             5        1 386 824        6 754
      Jihočeský kraj CZ031            17          637 047       10 570
        Ústecký kraj CZ042            19          798 898       15 030
        Zlínský kraj CZ072            19          572 432       10 407
Moravskoslezský kraj CZ080            24        1 177 989       27 100
     Pardubický kraj CZ053           164          514 518       84 040
    Karlovarský kraj CZ041         1 246          283 210      352 695
       Plzeňský kraj CZ032         6 654          578 707    3 850 164
     Liberecký kraj CZ051        11 728          437 570    5 131 707
   Jihomoravský kraj CZ064        12 966        1 184 568   15 358 712
```

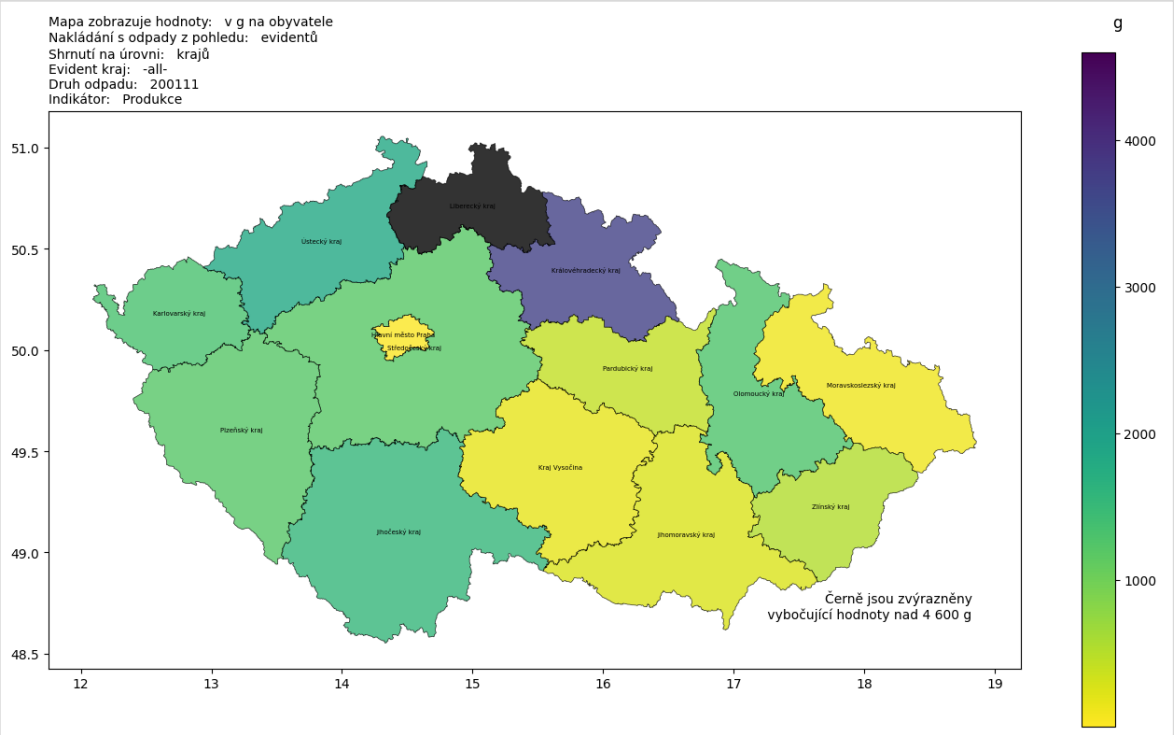
Obrázek 34. Zobrazení hodnot pro tvorbu mapy do text widgetu

Z DataFrame mapa_data jsou vyfiltrovány řádky, u kterých je hodnota ve sloupci ´Odpad_vKg´ > 0. Pro tyto řádky je dohledána informace o názvu území (Kraj_Nazev, ORP_Nazev, ZUJ_Nazev) v příslušném souboru s unikátními hodnotami pro kraje, ORP nebo ZÚJ, které vycházejí z Lexikonu obcí.

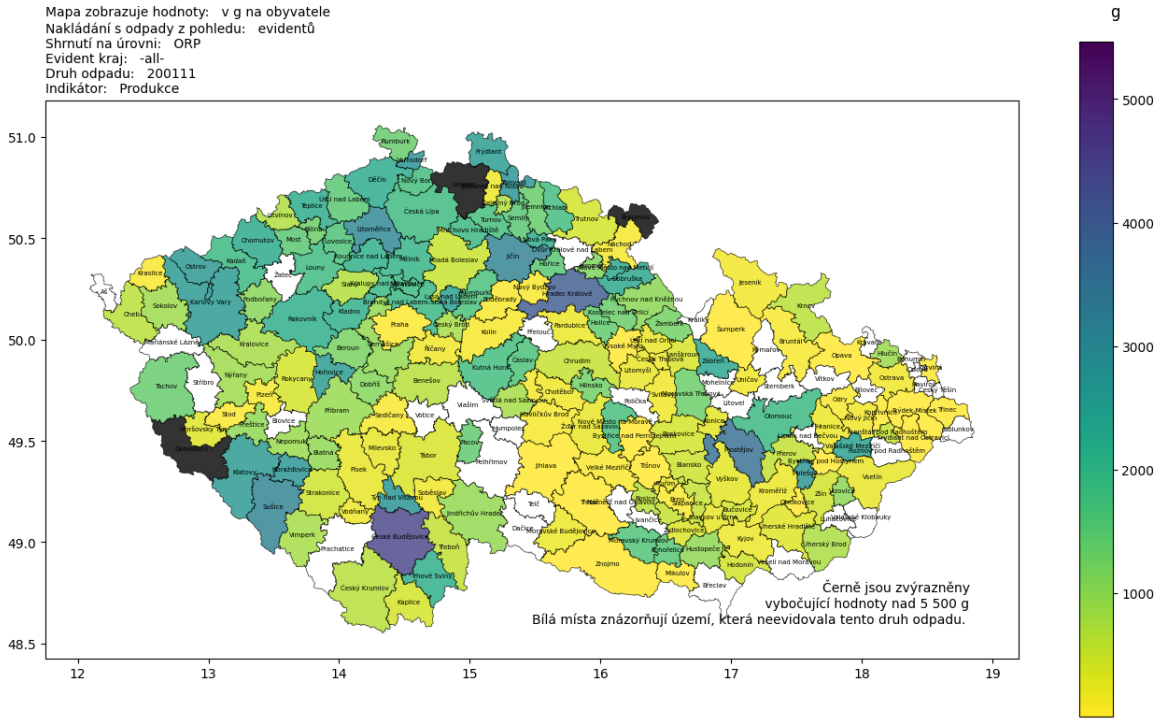
Dále je vytvořen seznam s názvy sloupečků, které se budou v text widgetu zobrazovat, data jsou formátována, seřazena a vložena do text widgetu.

7.5.1.3 Popis výsledné mapy jako nástroje pro analýzu dat

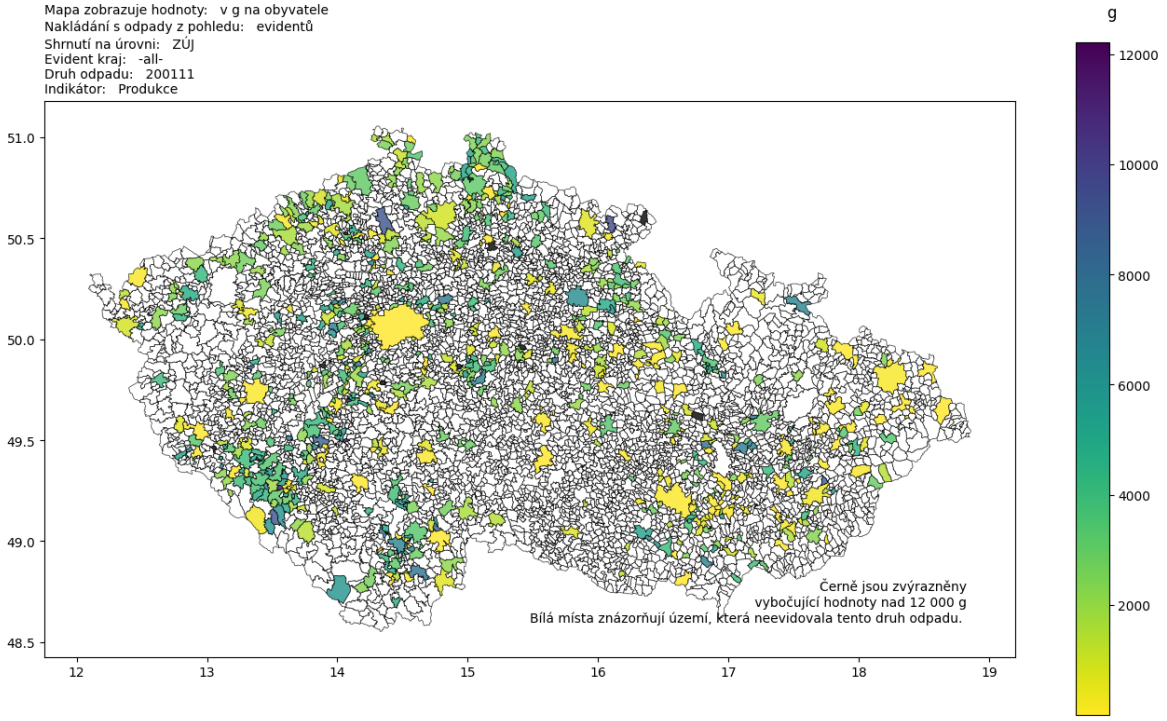
Ukázka výsledné mapy pro indikátor produkce odpadu 200111 (textilní materiál) v gramech na obyvatele pro různé geografické úrovně je uvedena na obrázcích: Obrázek 35, Obrázek 36 a Obrázek 37.



Obrázek 35. Mapa produkce textilního odpadu na obyvatele na úrovni krajů



Obrázek 36. Mapa produkce textilního odpadu na obyvatele na úrovni ORP

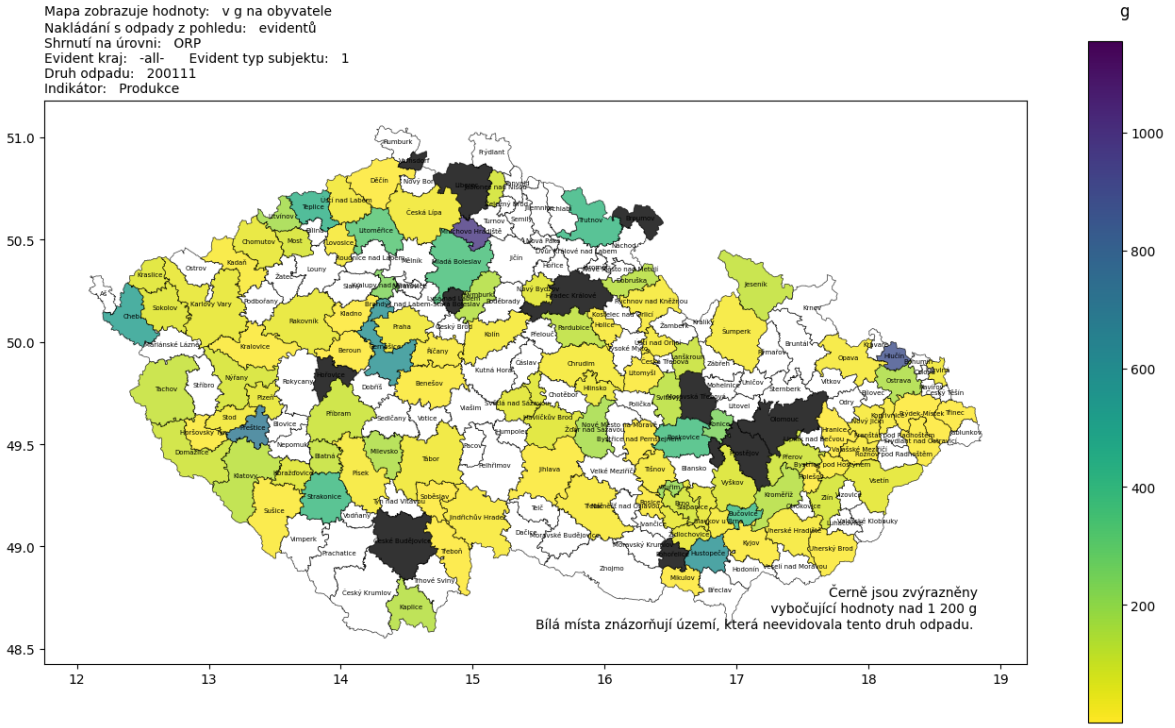


Obrázek 37. Mapa produkce textilního odpadu na obyvatele na úrovni ZÚJ

Použití funkcionality pro tvorbu mapy jako nástroje při statistickém zhodnocení dat lze spatřit v několika oblastech:

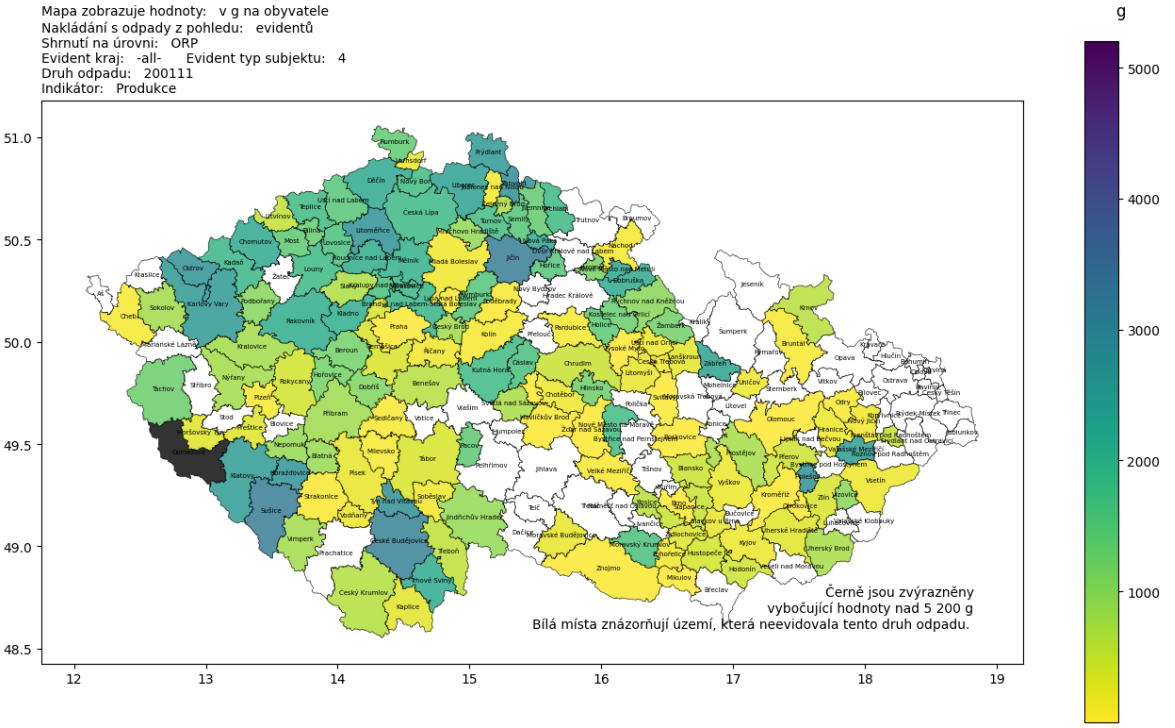
- Slouží pro rychlou vizualizaci dat. Na první pohled může uživatel vyčíst, jak se data rozložila geograficky. Jaké jsou hodnoty v jednotlivých oblastech.
- Mapa může pomoci identifikovat různé trendy v datech (vysoké či nízké hodnoty indikátorů, shlukování).
- Pomocí několika úrovňové mapy lze získat přístup k detailům (Kraj, ORP, ZÚJ). Zobrazit podrobnější informace o hodnotách v této oblasti.
- V případě dostupnosti dat za více období, může mapa pomoci s vizualizací změn v čase. Ukáže například jak se změnili hodnoty indikátorů v určité oblasti v průběhu času.

Lze třeba pozorovat změna v produkci textilního odpadu, pokud se nechá sestavit mapa na úrovni ORP zvlášť pro typ subjektu 1 – firmy (Obrázek 38) a pro typ subjektu 4 obce.



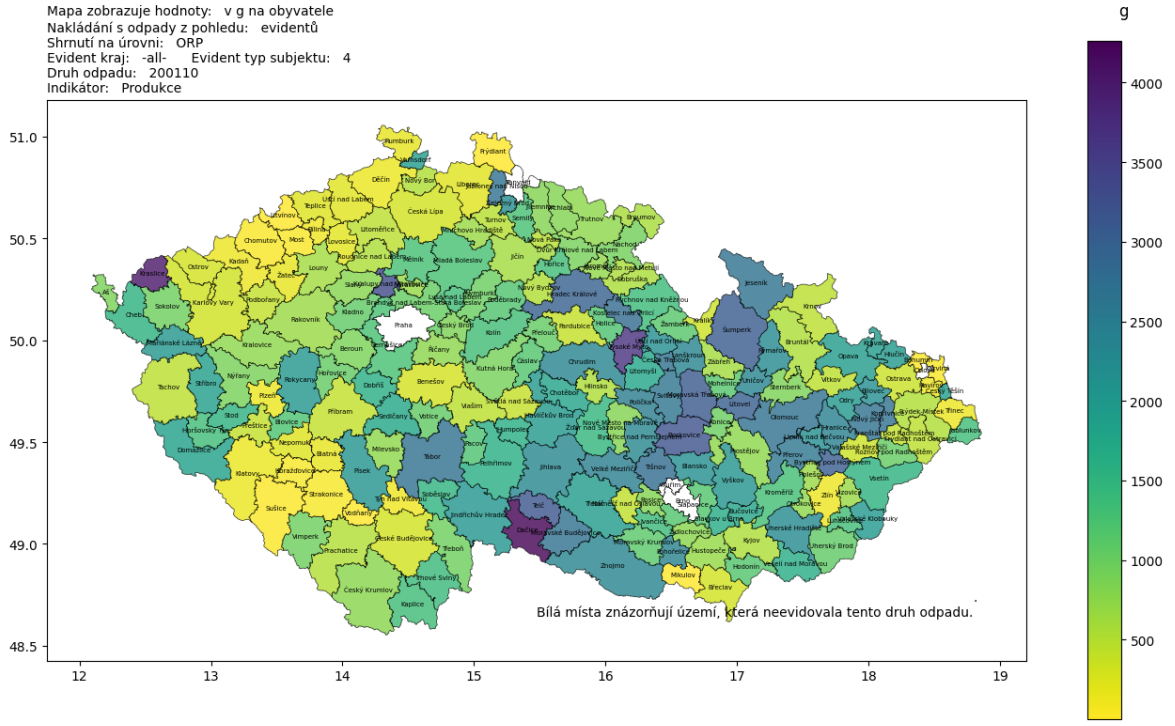
Obrázek 38. Mapa produkce textilního odpadu na obyvatele na úrovni ORP, evident typu 1 – firma

Při srovnání obou map je na první pohled vidět menší variabilita u obcí (Obrázek 39) a znatelné vyšší hodnoty odpadu pro severozápadní a jihozápadní část České republiky.



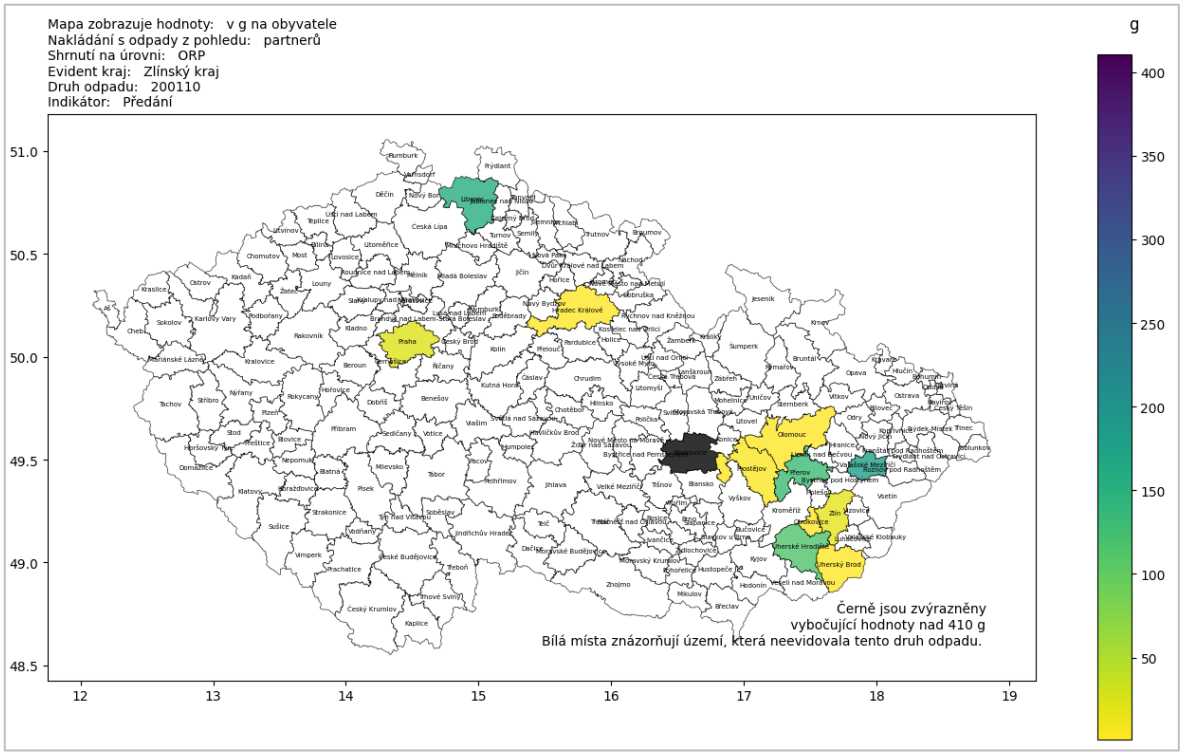
Obrázek 39. Mapa produkce textilního odpadu obcí na obyvatele na úrovni ORP

Pokud bychom srovnali s mapou pro produkci odpadu oděvy u obcí, budou zase viditelnější vyšší objemy v gramech na obyvatele na Moravě (Obrázek 40).



Obrázek 40. Mapa produkce odpadu oděvy obcí na obyvatele na úrovni ORP

Rovněž zajímavé je, když se v mapě nechá ukázat, na která místa předává svůj odpad oděvy 200 110 třeba Zlínský kraj (Obrázek 41). Zabarvený ORP Praha může poukazovat na možnou chybu ve vykazování odpadu, kdy se dle metodiky nemá vykazovat na sídlo firmy, ale na místo, kde odpad vznikl.



Obrázek 41. ORP na které Zlínský kraj předává odpad 200110 oděvy

7.5.2 Sumarizace dat

Pro úroveň krajů a ORP je připravena funkce *sumarizace()*, která shrnuje vlastnosti výběrového souboru, aby bylo snazší získat přehled o jeho charakteristikách. V této funkci jsou vypočítány: průměr, medián, směrodatná odchylka a jednotlivé kvartily (Obrázek 42), které jsou zobrazovány v textovém widgetu. Funkce je v celém znění uvedena v příloze (P XIII).

```
Charakteristiky polohy odpadu na obyvatele v jednotlivých krajích (g):

Indikátor: ['Produkce']   Kód nakládání: []
Druh odpadu: ['200111']   Rok: ['2021']
Kraj: []
ORP: []
```

Kraj	Pocet_hodnot	Prumer	Median	Minimum	25.percentil	50.percentil	75.percentil	Maximum	Smerodatna_odchylka
Hlavní město Praha	1	30	30	30	30	30	30	30	-
Jihomoravský kraj	78	1 913	1 342	5	317	1 342	3 162	6 979	1 858
Jihočeský kraj	72	3 182	2 591	2	1 119	2 591	5 173	11 690	2 604
Karlovarský kraj	17	2 589	2 159	115	820	2 159	4 185	8 281	2 194
Kraj Vysočina	14	1 487	1 470	14	406	1 470	2 162	3 381	1 158
Královéhradecký kraj	43	6 129	2 616	35	1 296	2 616	3 452	137 041	20 699
Liberecký kraj	58	72 766	2 821	112	2 139	2 821	3 974	4 052 534	531 739
Moravskoslezský kraj	17	1 171	167	0	30	167	2 167	6 851	1 847
Olomoucký kraj	35	3 096	2 986	1	858	2 986	4 501	8 038	2 403
Pardubický kraj	42	2 051	1 212	0	186	1 212	2 424	26 483	4 137
Plzeňský kraj	86	18 419	3 503	22	1 752	3 503	4 989	1 276 923	137 329
Středočeský kraj	150	4 353	2 785	0	1 249	2 785	4 268	154 543	12 761
Zlínský kraj	32	1 741	1 207	10	147	1 207	2 561	8 810	2 062
Ústecký kraj	67	2 480	2 149	33	1 257	2 149	3 040	12 210	2 090

Obrázek 42. Charakteristiky polohy vypočtené pro úroveň krajů

Funkce *sumarizace()* na svém vstupu přijímá globální proměnnou se vstupními daty. Dále se testuje, zda uživatel vybral některou ze ZÚJ z comboboxu v Evident frame, pokud ano, uživateli se vrátí do textového widgetu zpráva, že pro tuto úroveň nelze charakteristiky vypočítat a funkce se ukončí.

Pokud uživatel nevybral žádný kraj, ani ORP, je uživateli vrácena DataFrame s vypočítanými hodnotami za jednotlivé kraje (Obrázek 42). Jestliže uživatel zvolil některý kraj, nebo ORP, budou vypočítány charakteristiky na úrovni ORP (Obrázek 43).

```
Charakteristiky polohy odpadu na obyvatele v jednotlivých ORP (g):

Indikátor: ['Produkce']   Kód nakládání: []
Druh odpadu: ['200111']   Rok: ['2021']
Kraj: ['Jihočeský kraj']
ORP: []
```

ORP	Pocet_hodnot	Prumer	Median	Minimum	25.percentil	50.percentil	75.percentil	Maximum	Smerodatna_odchylka
Blatná	5	3 863	2 259	235	1 778	2 259	6 503	8 539	3 496
Jindřichův Hradec	3	1 935	1 663	1 429	1 546	1 663	2 188	2 713	683
Kaplice	2	2 982	2 982	1 337	2 160	2 982	3 804	4 627	2 326
Milevsko	3	1 361	1 302	343	822	1 302	1 871	2 439	1 049
Písek	4	5 976	6 093	27	4 059	6 093	8 011	11 690	4 796
Soběslav	2	2 598	2 598	11	1 304	2 598	3 891	5 184	3 657
Strakonice	4	1 274	1 060	790	905	1 060	1 428	2 185	627
Trhové Sviny	3	2 799	601	93	347	601	4 152	7 703	4 254
Tábor	4	1 782	1 791	70	508	1 791	3 065	3 476	1 672
Týn nad Vltavou	6	4 199	4 477	1 881	2 810	4 477	5 617	6 107	1 761
Třeboň	5	3 398	2 058	2	949	2 058	5 656	8 328	3 490
Vimperk	2	1 711	1 711	1 329	1 520	1 711	1 903	2 094	540
Vodňany	1	4 136	4 136	4 136	4 136	4 136	4 136	4 136	-
České Budějovice	22	3 559	3 286	65	1 982	3 286	5 183	7 446	2 338
Český Krumlov	6	2 695	2 496	136	706	2 496	4 367	5 940	2 389

Obrázek 43. Charakteristiky polohy vypočtené pro úroveň ORP

Volby uživatele slouží jako parametry na základě, kterých se určují sloupce, ze kterých se výsledný DataFrame bude dále zpracovávat (ORP či kraj). Vytvoří se prázdná DataFrame, která slouží k ukládání vypočtených charakteristik pro jednotlivá území. Ze vstupního datasetu se zjistí unikátní hodnoty pro názvy krajů nebo ORP a seřadí se abecedně. Ve for cyklu se pro každé území (kraj nebo ORP) vyberou hodnoty ze sloupce ´odpadNaObyv_g´ a uloží se do seznamu odpad_seznam. Následně se vytvoří seznam s názvem charakteristi-

ky. V rámci for cyklu se pro každý datový soubor ze seznamu odpady_seznam vypočítají jednotlivé charakteristiky. Výpočty se následně přidají do seznamu charakteristiky. Z tohoto seznamu se vytvoří DataFrame s názvem metriky_df.

DataFrame metriky_df se uloží do globální proměnné výsledek_excel, kterou je možné uložit do XLSX souboru stisknutím tlačítka v GUI Uložit do xlsx.

V závěru funkce se vytvoří textový widget. Sloupce výsledné DataFrame se formátují a spolu s vypsanými volbami uživatele se zobrazují v text widgetu.

7.5.3 Seskupení dat

Pokud uživatel potřebuje získat přehled o množství odpadu v kg při různém seskupení dat, může využít funkci s názvem Seskupení dat. Funkce je dostupná z výběrového combo boxu v GUI a do textového widgetu vypisuje tabulku se seskupenými daty.

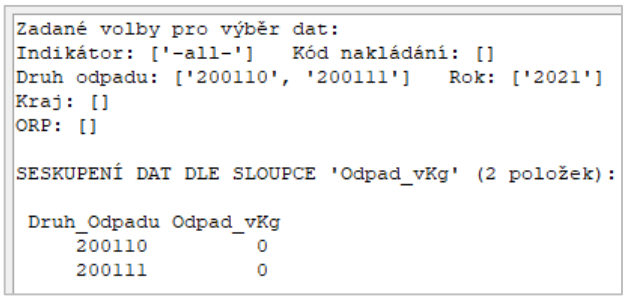
Sloupce, podle kterých chce uživatel data seskupit, se vybírají z combo boxu pojmenovaného Sloupce k seskupení. Uživatel vybere tolik sloupců, kolik potřebuje (Obrázek 44). V případě, že nechce sloupce vybírat sám, použije se automaticky list s již navolenými názvy sloupců. Po výběru sloupců uživatel vybere funkci Seskupení dat a funkci spustí stiskem tlačítka Spuštění funkce.

```
Zadané volby pro výběr dat:
Indikátor: ['Skládkování']   Kód nakládání: []
Druh odpadu: ['200110', '200111']   Rok: ['2021']
Kraj: []
ORP: []

SESKUPENÍ DAT DLE SLOUPCE 'Odpad_vKg' (12 položek):

   Evident_Kraj_Nazev  Odpad_vKg
   Jihomoravský kraj      -9 590
      Jihočeský kraj    -287 817
    Karlovarský kraj     -33 200
Královéhradecký kraj -1 457 101
      Liberecký kraj     -15 451
Moravskoslezský kraj    -165 638
      Olomoucký kraj    -668 283
     Pardubický kraj     -98 046
       Plzeňský kraj     -52 300
    Středočeský kraj    -139 589
        Zlínský kraj     -33 370
        Ústecký kraj    -185 889
```

Obrázek 44. Seskupení dat podle sloupce Evident_Kraj_Nazev pro indikátor skládkování

Na textovém widgetu se zobrazí nejprve volby uživatele pro specifikaci kritérií při tvorbě výběrového souboru a poté tabulka se seskupenými daty. Při seskupení se používá funkce suma pro sloupec ´Odpad_vKg´. Aby bylo patrno, zda způsob nakládání s odpady působí

kladně či záporně v ročním zúčtování, jsou ponechána u hodnot znaménka +/-. Pokud bychom chtěli udělat například seskupení dat za jednotlivé druhy odpadů, které jsou v modelu nahrány, zjistíme, že součet je u každého odpadu 0, což je dáno právě kladným či záporným působením jednotlivých kódů nakládání v ročním zúčtování. Může nám to sloužit jako kontrola, že data jsou v tomto ohledu správná (Obrázek 45).



Obrázek 45. Kontrola nulové roční bilance pro jednotlivé druhy odpadů

Výběry a seskupení dat jsou v programovém řešení realizovány prostřednictvím funkce *seskupeni_dat_seznam_sloupcu()* (Program 29), která je spouštěna funkcí *seskupeni_dat()*.

```
def seskupeni_dat_seznam_sloupcu(data, func_column,
group_column_list ):
    grouped_data = da-
ta.groupby(group_column_list)[func_column].sum().reset_index()
    return grouped_data
```

Program 29. Funkce *seskupeni_dat_seznam_sloupcu()* použití při seskupování dat

Funkce používá metodu *.groupby()* na DataFrame, která podle seznamu sloupců seskupí řádky. Ve sloupci ´Odpad_vKg´ je provedena agregační funkce *sum()*, která sečte množství kilogramů. Funkce vrací DataFrame vysledek.

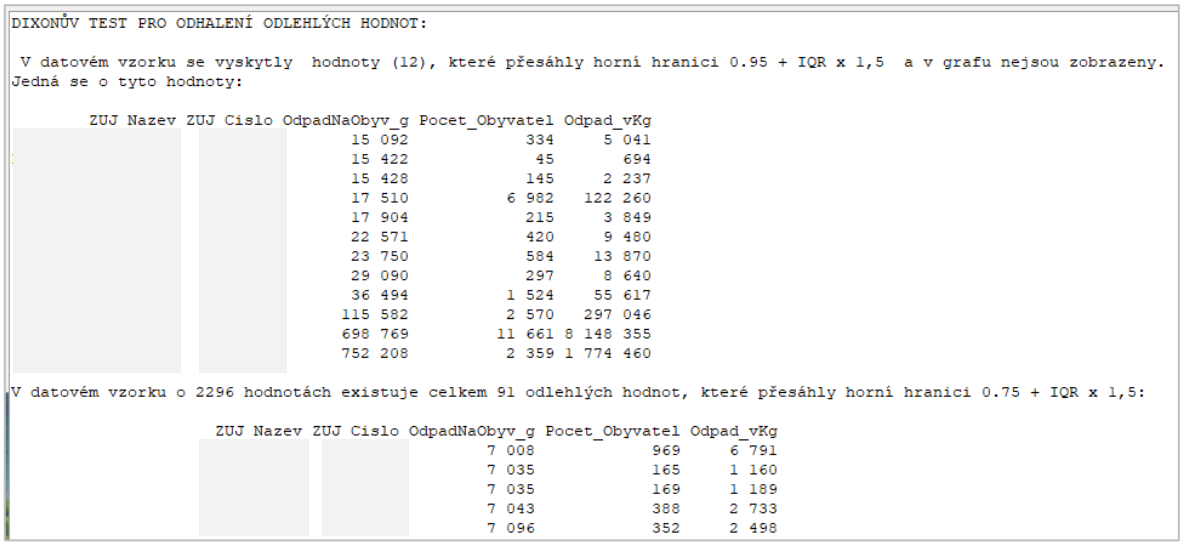
7.5.4 Zjištění odlehlých hodnot

Odlehlé hodnoty jsou datové body, které se podstatně liší od ostatních. Cílem funkcionality bude nalézt tyto datové body, zobrazit je uživateli v textovém widgetu a vizualizovat hodnoty odpadu na obyvatele v diagramu rozptýlení. Řešení poskytuje funkce *odlehle_hodnoty()* a v plném znění je uvedena v příloze (P XII).

7.5.4.1 Popis programového řešení funkce odlehle_hodnoty()

Funkce *odlehle_hodnoty()* na svém vstupu načítá globální proměnnou vyber_dat_vysledek, která v sobě obsahuje výběrový soubor. Z výběrového souboru se získá DataFrame odpad_obyvatele, kde se vypočítá odpad na obyvatele v gramech. Podrobně je popsáno v kapitole 7.5.6.1.

Pomocí funkce *zjisteni_hranic(df, columns)* získáme hranice lower_bound a upper_bound, které budou sloužit pro vymezení DataFrame outliers. Ta bude obsahovat takové datové body, které jsou menší než hranice lower_bound a větší než hranice upper_bound. To, jak zjistit hranice pro určení odlehlých hodnot je uvedeno v kapitole 7.5.4.3.. Hodnoty q1 a q3 jsou vloženy do funkce jako proměnné a jsou nastaveny na hodnoty 0.25 a 0.75. DataFrame outliers je následně zobrazena v textovém widgetu (Obrázek 46).

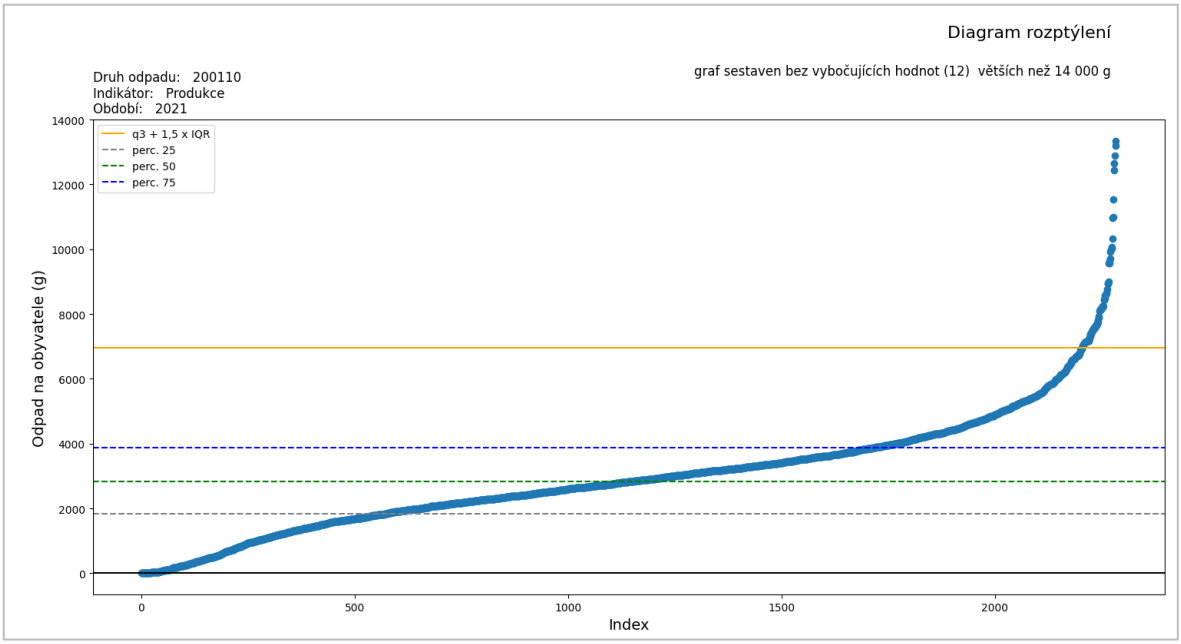


```
DIXONŮV TEST PRO ODHALENÍ ODLEHLÝCH HODNOT:

 V datovém vzorku se vyskytly  hodnoty (12), které přesáhly horní hranici 0.95 + IQR x 1,5  a v grafu nejsou zobrazeny.
Jedná se o tyto hodnoty:

       ZUJ Nazev ZUJ Cislo OdpadNaObyv_g Pocet_Obyvatel Odpad_vKg
                                 15 092            334     5 041
                                 15 422             45       694
                                 15 428            145     2 237
                                 17 510          6 982   122 260
                                 17 904            215     3 849
                                 22 571            420     9 480
                                 23 750            584    13 870
                                 29 090            297     8 640
                                 36 494          1 524    55 617
                                115 582          2 570   297 046
                                698 769         11 661 8 148 355
                                752 208          2 359 1 774 460

V datovém vzorku o 2296 hodnotách existuje celkem 91 odlehlých hodnot, které přesáhly horní hranici 0.75 + IQR x 1,5:

                  ZUJ Nazev ZUJ Cislo OdpadNaObyv_g Pocet_Obyvatel Odpad_vKg
                                              7 008            969     6 791
                                              7 035            165     1 160
                                              7 035            169     1 189
                                              7 043            388     2 733
                                              7 096            352     2 498
```

Obrázek 46. Textový widget pro výpis odlehlých hodnot

Jelikož výběrové soubory obsahují tak odlehlé hodnoty, které nedovolují zcela přehledně zobrazit datové body v diagramu rozptýlení. Je proveden výpočet odlehlých hodnot ještě před samotným zjišťováním outliers s parametry 0.15 a 0.95. Tím získáme několik málo datových bodů, které způsobovaly nečitelnost grafu. Tyto datové body jsou rovněž uvedeny v textovém widgetu (Obrázek 46) a uživatel je upozorněn na to, že nejsou součástí datové sady, která je v diagramu rozptýlení vizualizována. Diagram rozptýlení je v knihovně matplotlib konstruován jako bodový diagram scatter plot. Do grafu jsou přidány linie vodorovné s osou Y znázorňující 25., 50. a 75. percentil. Oranžově je vložena linie s hranicí upper_bound, která je vypočítána jako hodnota třetího kvartilu, ke které je přičten 1,5násobek vzdálenosti mezi 25. a 75. percentilem.

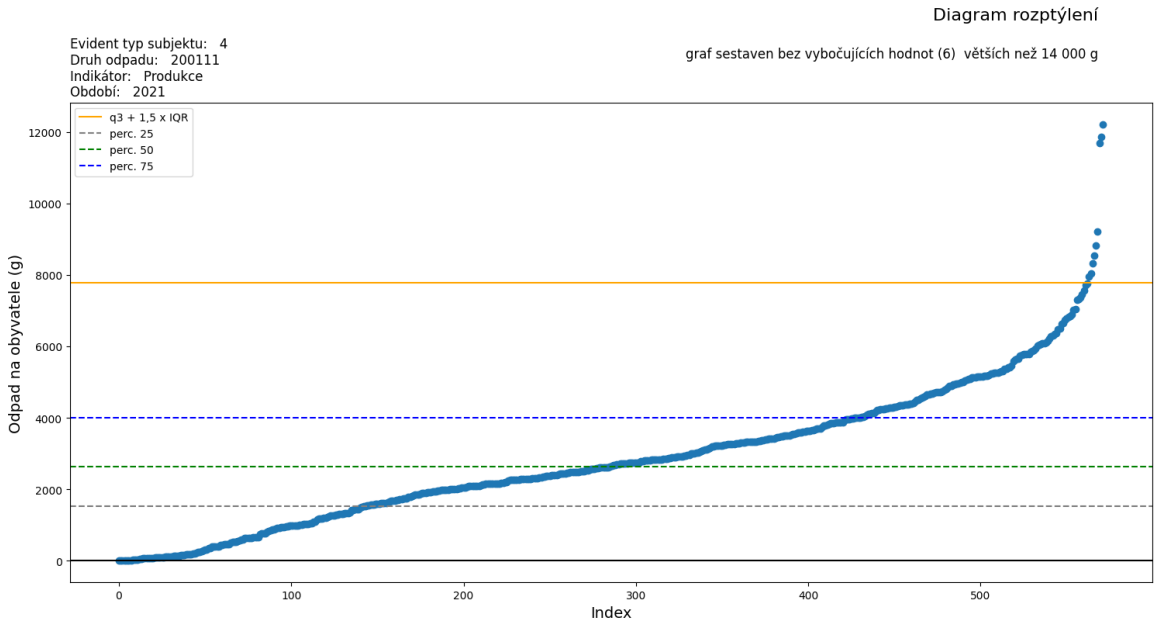
Graf je opatřen legendou vodorovných linií, popisky specifikujícími volby uživatele pro sestavení výběrového souboru v levém horním rohu plochy grafu a v pravém horním rohu je uveden název grafu s informací o tom, zda byly z datasetu odstraněny vybočující hodnoty (Obrázek 47).



Obrázek 47. Diagram rozptýlení pro datové body produkce odpadu oděvy

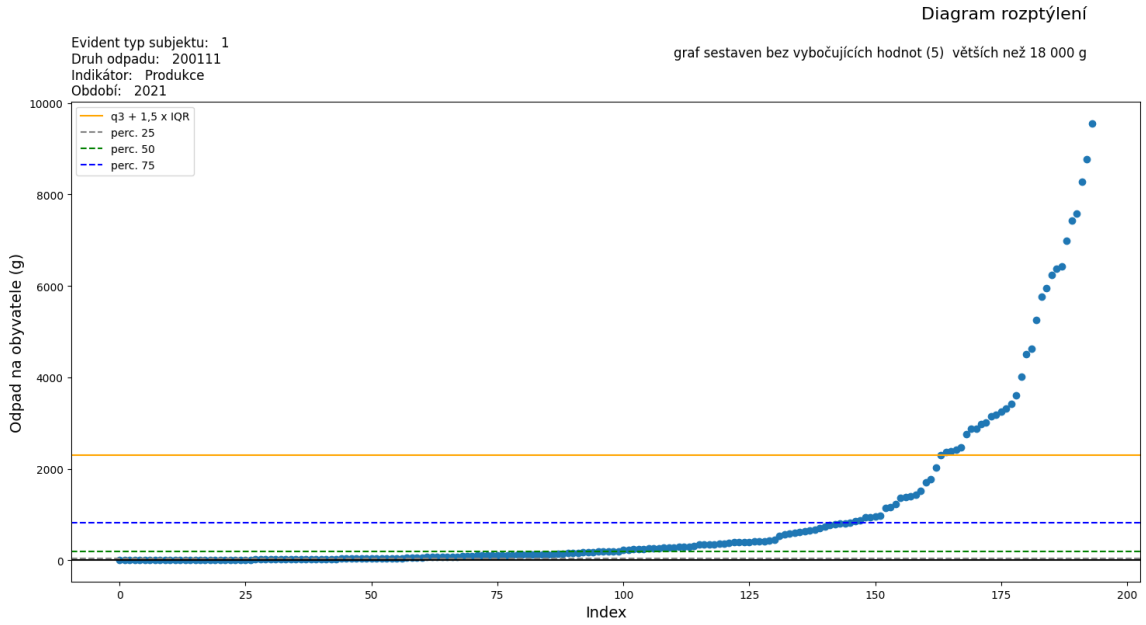
7.5.4.2 Popis diagramu rozptýlení jako nástroje pro analýzu dat

Diagram rozptýlení je bodový graf, ve kterém jsou data seřazena od nejmenšího po největší. Na ose X se nachází index datového bodu a na ose Y jsou vyneseny hodnoty datových bodů, tj. odpad na obyvatele v gramech. Datové body vykreslují křivku, která dle svého tvaru, strmosti a hustoty bodů napovídá o tom, jak jsou data rozptýlena v datasetu. Rozdíl můžeme vidět, pokud srovnáme grafy sestavené pro produkci odpadu textilního materiálu pro evidenty typu 4 – obce (Obrázek 48) a evidenty typu 1 – firmy (Obrázek 49).



Obrázek 48. Diagram rozptýlení pro produkci textilního odpadu obcí

Tvary křivek, které jsou vytvořeny z datových bodů, jsou zcela odlišné. To, kdy křivka překonává hranici 25. a 50. percentilu se odehrává u obcí na úrovni kolem 2,5 kg, kdežto firmy 50. percentil překonávají s hodnotou kolem 0,3 kg odpadu. Zhruba každá 6. obec vykazuje odpad, kdežto u firem vykazuje pouze třetinové množství. U firem je zanedbatelné vyprodukované množství pro většinu vykazovaných územních jednotek. A pouze několik ZÚJ vykazuje v případě firem vyšší produkci. V případě odlehlých hodnot se obce a firmy chovají podobně, což je zajímavé. Může to znamenat, že v některých obcích existují subjekty vlastněné obcí, které v odpadovém odvětví textilu vykazují značnou činnost. Mohly by to být třeba obce, které zřídily společnost pro vybírání odpadu.



Obrázek 49. Diagram rozptýlení pro produkci textilního odpadu u firem

7.5.4.3 Funkce zjisteni_hranic() pro zjištění vybočujících datových bodů

Ve funkcionalitách, které dále v práci slouží pro analýzu dat, se často vyskytuje problém s existencí vybočujících hodnot, které znemožňují dostatečně vizualizovat data v grafech. Proto jsou z výběrového souboru vyloučeny řádky, které mají hodnotu ve sloupci ´OdpadNaObyv_g´ vyšší než je horní hranice vypočítaná pomocí funkce *zjisteni_hranic(df, columns)*. Pro určení horní hranice se využívá pravidlo, že jakýkoli datový bod větší než 85. percentil plus 1,5násobek IQR je považován za vybočující. Mezikvartilový rozsah IQR se vypočítá jako rozdíl mezi 15. a 85. percentilem (Program 30).

```
def zjisteni_hranic(df, columns):
    q1 = df[columns].quantile(0.15)
    q3 = df[columns].quantile(0.85)
    iqr = q3 - q1
    spodni_hranice = q1 - 1.5 * iqr
    horni_hranice = q3 + 1.5 * iqr
    return spodni_hranice, horni_hranice
```

Program 30. Funkce *zjisteni_hranic()* pro stanoveni hranice pro vybočující datové body

7.5.5 Relativní četnosti

Pro získání představy o tom, jak často se vyskytuje určitý jev v souboru, je možné použít relativní četnosti. V případě vyhodnocení dat z odpadového hospodářství bude za tento jev

považováno počet gramů odpadu na obyvatele zkoumaného územního celku při zvoleném způsobu nakládání s odpadem. A další specifické volby uživatele pro sestavení výběrového souboru (druh odpadu, období, území evidenta, typ subjektu).

Výpočet relativních četností řeší funkce *relativni_cetnosti()*, která je v celém svém znění uvedena v příloze (P IX). Cílem funkce je vypočítat relativní četnosti, zobrazit výsledky v textovém widgetu a vykreslit sloupcový graf. Na začátku funkce je načítána globální proměnná vyber_dat_vysledek. Je testováno, zda obsahuje data a pokud ne, je funkce ukončena s výpisem informace pro uživatele. Proměnná přináší Data Frame s výběrovým souborem, který vznikl dle zadání uživatele prostřednictvím funkce *vyber_evident_partner_kriteria()*.

Následují podmínky, které zjišťují, jaké volby uživatel zvolil pro to, aby se správně nastavili zdrojové soubory a proměnné použité dále v kódu. Podle toho, jakou geografickou úroveň uživatel zvolil, tak se bude provádět výpočet množství odpadu na obyvatele. Sečte se odpad za konkrétní území, ale dělit se bude počtem obyvatel zadaným v souboru Lexikonu obcí nebo v Unique_kraj či Unique_ORP. Výpočet se vkládá jako nový sloupec do DataFrame výsledek a objeví se i v text widgetu jako sloupec ´OdpadNaObyv_g´.

Pro výpočet četností se sčítají hodnoty ve sloupci ´OdpadNaObyv_g´ do proměnné soucet a následně se každý řádek podělí tímto součtem. Výpočet se vloží do nového sloupce ´Relativni_cetnost´. Na tento sloupec se aplikuje změna formátu na procenta. Výsledná tabulka se zobrazí v text widgetu (Obrázek 50) a rovněž si ji uživatel může uložit do MS Excelu prostřednictvím tlačítka Uložit do xlsx.

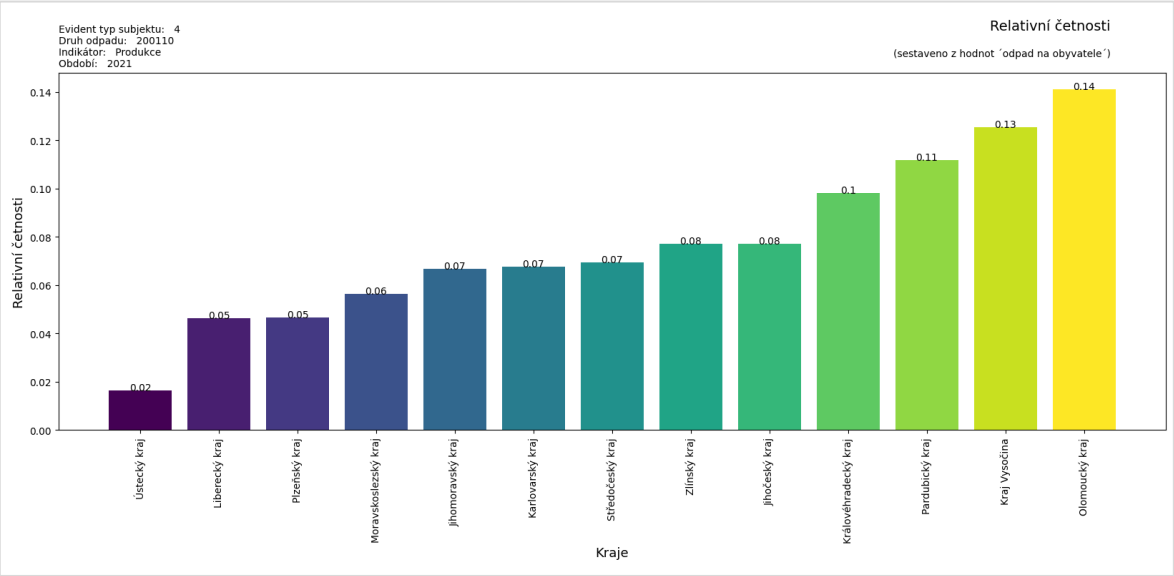
Procentuelní zastoupení jednotlivých krajů:

Kraj_Nazev	Kraj_Cislo	OdpadNaObyv_g	Pocet_Obyvatel	Odpad_vKg	Relativni_cetnost
Ústecký kraj	CZ042	250	798 898	200 019	1.65%
Liberecký kraj	CZ051	705	437 570	308 654	4.64%
Plzeňský kraj	CZ032	707	578 707	409 604	4.66%
Moravskoslezský kraj	CZ080	858	1 177 989	1 011 458	5.65%
Jihomoravský kraj	CZ064	1 013	1 184 568	1 200 264	6.67%
Karlovarský kraj	CZ041	1 028	283 210	291 351	6.77%
Středočeský kraj	CZ020	1 056	1 386 824	1 464 715	6.95%
Zlínský kraj	CZ072	1 169	572 432	669 368	7.70%
Jihočeský kraj	CZ031	1 170	637 047	745 654	7.70%
Královéhradecký kraj	CZ052	1 488	542 583	807 750	9.80%
Pardubický kraj	CZ053	1 697	514 518	873 237	11.17%
Kraj Vysočina	CZ063	1 904	504 025	959 896	12.54%
Olomoucký kraj	CZ071	2 140	622 930	1 333 575	14.09%

Obrázek 50. Výpis v text widgetu funkce *relativni_cetnosti()*, úroveň krajů

Výsledky se zobrazují dle zvolených geografických úrovní pomocí comboboxů v Evident frame. Pokud zůstanou všechny comboboxy nevybrány, provedou se výpočty na úrovni krajů. Pokud se zvolí kraj či kraje, ve výsledné tabulce a grafu budou zobrazena data pro ORP pod které kraje spadají. A jestliže uživatel zvolí některé z ORP, tak se vypočítají hodnoty pro ZÚJ.

Vypočtené relativní četnosti jsou následně zobrazeny ve sloupcovém grafu, na který je aplikována color map viridis (Obrázek 51). Ke grafu je v levém horním rohu připojen titulek, který popisuje kritéria volby výběrového souboru uživatelem.



Obrázek 51. Na úrovni krajů graf zobrazuje relativní četnosti produkce odpadu oděvů na obyvatele pro evidenty typu 4 tj. obce

7.5.6 Histogram četností

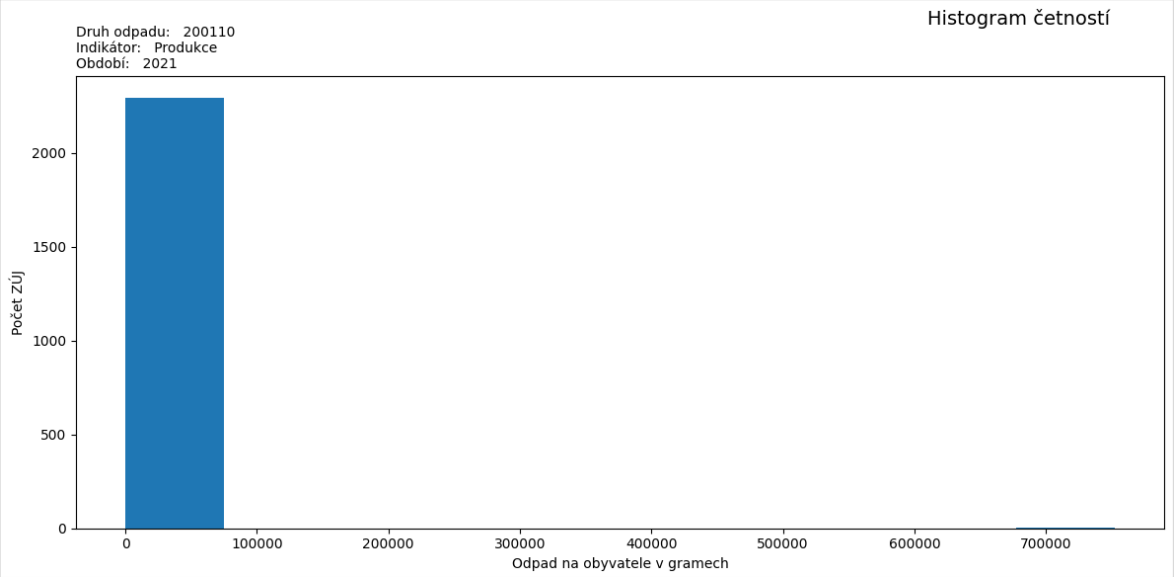
Pro získání přehledu o rozdělení hodnot odpad na obyvatele ve výběrovém souboru byla připravena funkce *histogram()*. Kód funkce je uveden v příloze (P X). Funkce nejdříve do textového widgetu zobrazí data, ze kterých je následně sestaven graf v podobě histogramu. Vstupem do grafu jsou hodnoty odpadu na obyvatele v gramech za jednotlivé ZÚJ.

7.5.6.1 Popis programového řešení histogramu

Na začátku funkce je načtena globální proměnná vyber_dat_vysledek, která předává vstupní DataFrame s výběrovým souborem. U dat ze vstupního souboru se nejprve v rámci funkce *odpad_naObyvatele_g()* provede sloučení podle hodnot ve sloupci ´Evident_ZUJ_Cislo´ a vypočte se množství odpadu v každé skupině. Poté se stejným způ-

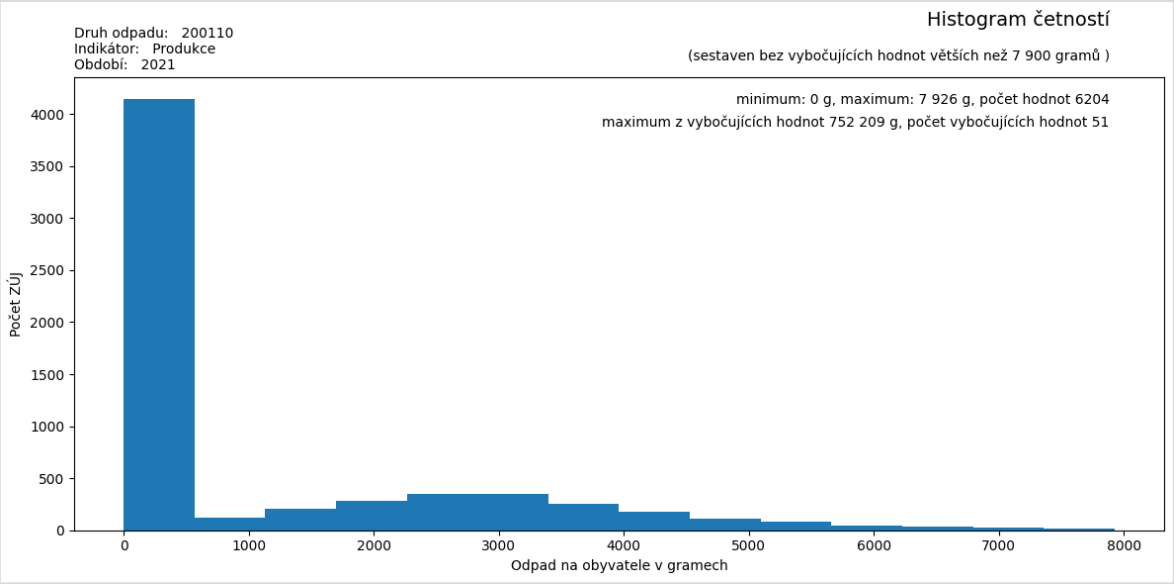
sobem seskupí údaje v tabulce Lexikon obcí rovněž podle sloupce ´ZUJ_Cislo´ a navzájem se DataFrame spojí do jedné DataFrame odpad_obyvatele. Zde se potom vypočítá odpad na obyvatele a vynásobí 1 000 abychom získali hodnoty v gramech.

V případě, že bychom vytvořili histogram z takto připravené DataFrame, mohli bychom získat graf, který v důsledku existence vybočujících hodnot způsobí, že většina hodnot bude v prvním intervalu a vybočující hodnoty se budou vyskytovat v posledním intervalu. Tak jak je to patrno v grafu na obrázku (Obrázek 52).



Obrázek 52. Histogram s přítomností vybočujících hodnot

Další problém se zobrazením dat v histogramu způsobuje fakt, že ne všechny ZÚJ zvolený odpad v daném období v ročním zúčtování vykázaly, a ve výsledné DataFrame se objevují s hodnotou 0. V příkladě na obrázku (Obrázek 53) se to týká celkem 3 963 ZÚJ.



Obrázek 53. Histogram s přítomností nulových hodnot u ZÚJ, které nevykázaly odpad v daném období

Všechny ZÚJ s nulovými hodnotami jsou tedy zahrnuty do prvního intervalu a díky tomu pak rozložení zbylých 2 296 ZÚJ není v histogramu zcela dobře viditelné.

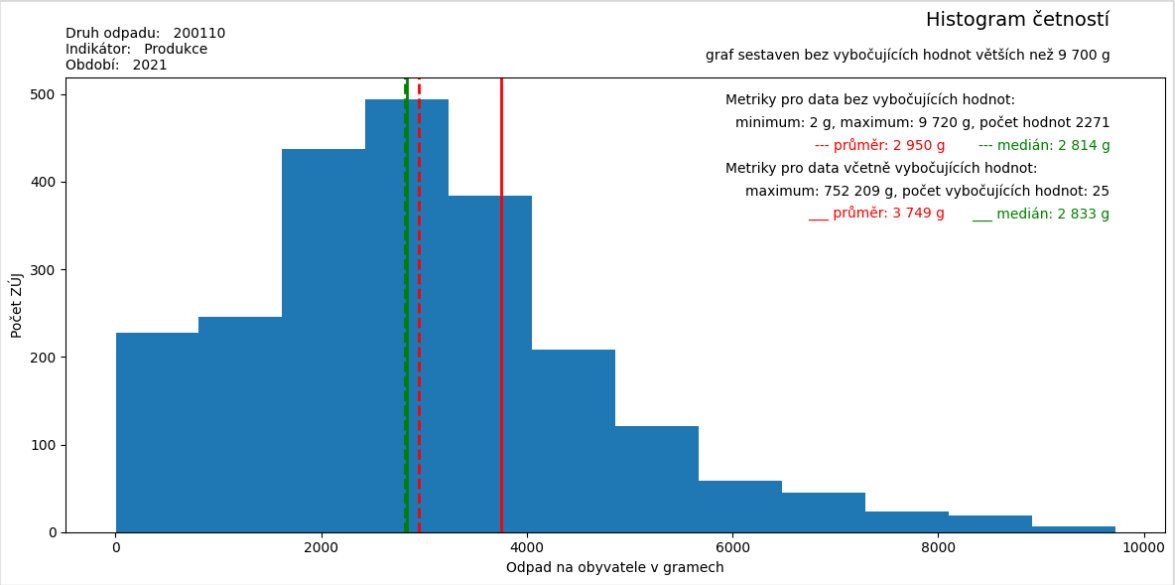
Tento důvod vedl k tomu, že během slučování při vzniku DataFrame odpad_obyvatele nebude použit způsob left join, ale inner join. A aby histogram lépe vizualizoval rozložení hodnot, budou z datasetu vyloučeny hodnoty, které jsou nad horní hranicí, jež se zjišťuje při výpočtu odlehlých hodnot. Toto zajišťuje funkce *zjisteni_hranic(df, columns)*, která je blíže popsána v kapitole 7.5.4.3.

Při stanovování počtu intervalů pro vykreslení histogramu je využito Sturgessovo pravidlo. Výsledkem je proměnná k, která slouží jako parametr pro počet intervalů (bins) (Program 31).

```
vysledek_lower = vysledek[vysledek['OdpadNaObyv_g'] <=
horni_hranice]
pocty_lower = len(vysledek_lower)
n = pocty_lower
k = round(1 + 3.322 * math.log10(n))

fig, ax = plt.subplots(figsize=(12,6))
plt.hist(vysledek_graf['OdpadNaObyv_g'],bins=k)
plt.xlabel('Odpad na obyvatele v gramech')
plt.ylabel('Počet ZÚJ')
```

Program 31. Stanovení počtu intervalů v histogramu pomocí Sturgessova pravidla

Výsledný graf je opatřen popisky Obrázek 54. V levém horním rohu je vypsána specifikace tvorby výběrového souboru a v pravém horním rohu se nachází informace o tom, zda byly



Obrázek 54. Histogram zobrazující rozložení produkce odpadu oděvy (200110) na obyvatele

při sestavení histogramu odstraněny vybočující hodnoty. Také je zde zvlášť, pro data bez vybočujících hodnot a zvlášť pro data včetně vybočujících hodnot, vypočítáno minimum, maximum, průměr, medián a počet hodnot.

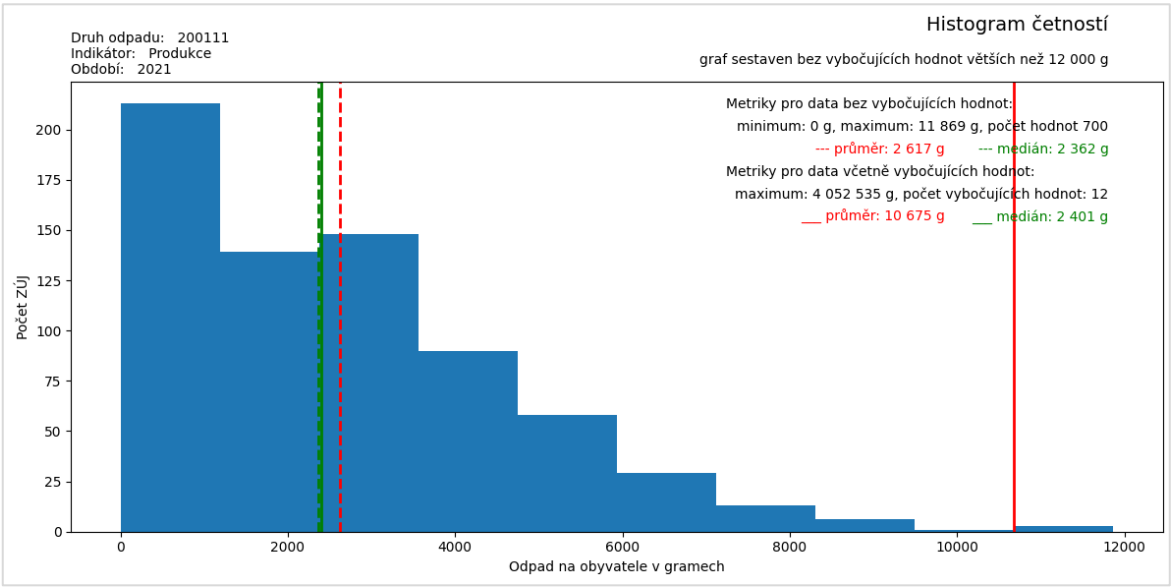
Průměr a medián jsou přidány do grafu jako vertikální čáry na zadaných hodnotách na x-ové ose. V kódu je to realizováno použitím metody *ax.axvline()*. Do grafu je přidána červená přerušovaná čára pro průměr a zelená přerušovaná čára pro medián pro dataset ze kterého jsou vynechány vybočující hodnoty. Pro dataset, který obsahuje celý výběrový soubor jsou tyto čáry plné, barvy zůstávají stejné jako v předchozím popisu.

Aby měl uživatel přehled, z jakých dat je histogram sestaven, jsou tato data zobrazena v textovém widgetu. Pokud dataset obsahoval vybočující hodnoty, je připravena zvláštní tabulka právě s těmito hodnotami hned na začátku text widgetu (v programu se jedná o DataFrame vysledek_higher). Po ní následuje tabulka bez vybočujících hodnot, ta je v programu uložena jako DataFrame vysledek_lower. Pokud se vybočující hodnoty nevyskytují, je rovnou zobrazena DataFrame vysledek_lower.

7.5.6.2 Popis histogramu jako nástroje pro analýzu dat

Histogram se vytvořil tak, že se vzaly hodnoty odpadu na obyvatele pro jednotlivé ZÚJ a rozdělili se do intervalů. Výšky pruhů představují počet ZÚJ, které v ročním zúčtování vykázaly odpad na obyvatele z daného intervalu. Pokud se podíváme na obrázek (Obrázek 54) tak zhruba 230 ZÚJ vykázalo produkci odpadu oděvů v intervalu $\langle 0\ ;\ 808 \rangle$ gramů.

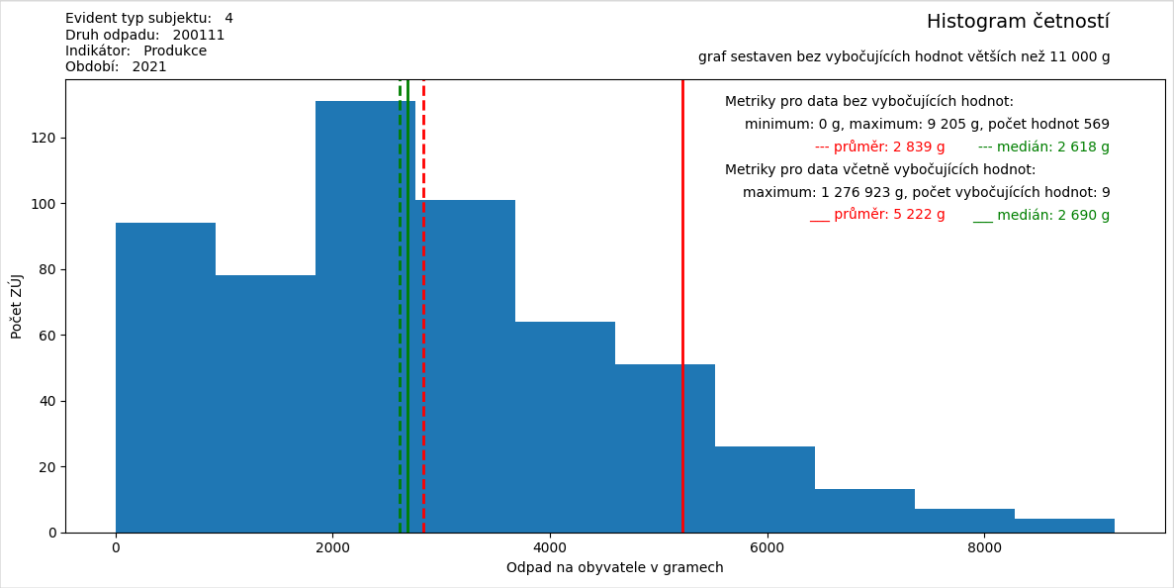
Z histogramu můžeme snadno získat představu o distribuci hodnot, o tom, jak se hodnoty rozkládají od nejmenší po největší. Říká nám, jak často se vyskytuje určitý rozsah hodnot a jaký je rozptyl mezi nimi. Tvar histogramu nám může naznačovat normální rozdělení, asymetrické rozdělení nebo přítomnost vybočujících hodnot. Rozdílnost tvarů histogramů můžeme pozorovat, pokud srovnáme produkci odpadu oděvy 200110 (Obrázek 54) a produkci textilního odpadu 200111 (Obrázek 55). Histogram pro textilní odpad znázorňuje asymetrické rozdělení. Data jsou nahromaděná nalevo s ocasem napravo. Jedná se tedy o data se zkreslením doprava. Když jsou data zkreslená, tak se průměr a medián liší. Průměr je posunut ve směru zkreslení doprava. Protože je průměr tažen vybočujícími hodnotami, je lepší pro zjištění středu použít medián, protože je méně ovlivněn těmito hodnotami. Průměr je citlivější na extrémní hodnoty a funguje lépe pro symetrická data.



Obrázek 55. Histogram s produkcí textilního odpadu se zkreslením doprava

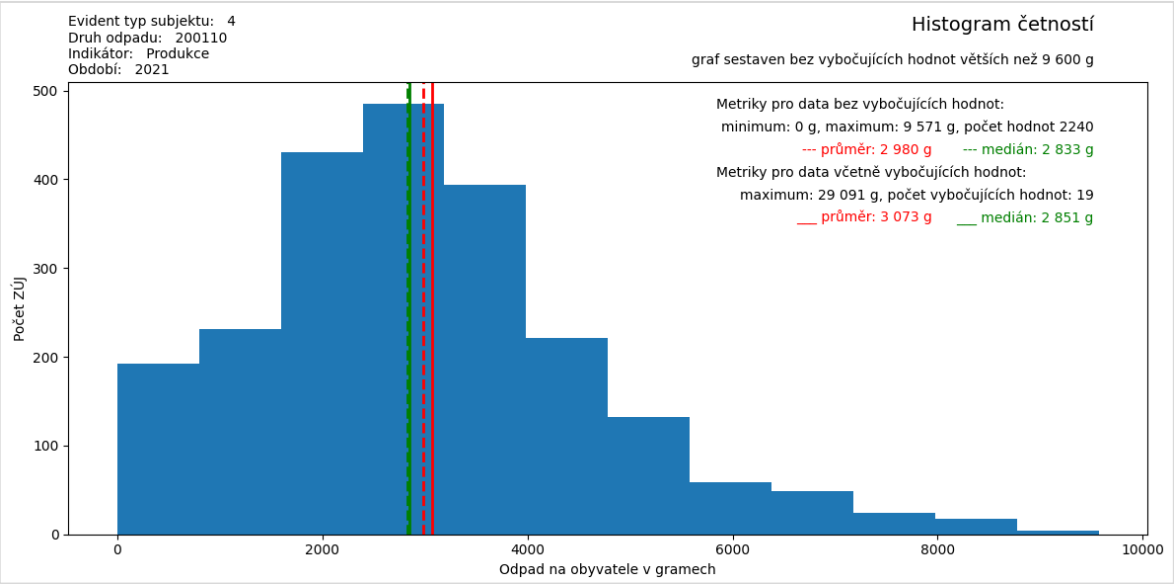
To dokazuje i pokud srovnáme průměr a medián u datasetu včetně vybočujících hodnot a bez vybočujících hodnot (Obrázek 55). Vybočující hodnoty táhnou průměr až na úroveň 10 657 gramů, kdežto mediány jsou si v obou datasetech velmi blízké na hodnotě 2 362 a 2 401 g.

Pokud bychom sestavily histogramy pouze pro produkci odpadů pro obce (typ subjektu 4) zjistíme, že medián i průměr se více k sobě přiblížili u obou sledovaných druhů odpadů (Obrázek 56).



Obrázek 56. Histogram pro produkci textilního materiálu pouze u obcí

U oděvů jsou dokonce velmi blízké výsledky pro oba datasety (tj. data bez i data s vybočujícími hodnotami) (Obrázek 57). Z toho můžeme usoudit, že při použití výběrového souboru pouze s typem subjektu 4, obsahují data méně odlehlých hodnot a data jsou symetrická.



Obrázek 57. Histogram pro produkci odpadu oděvy pouze u obcí

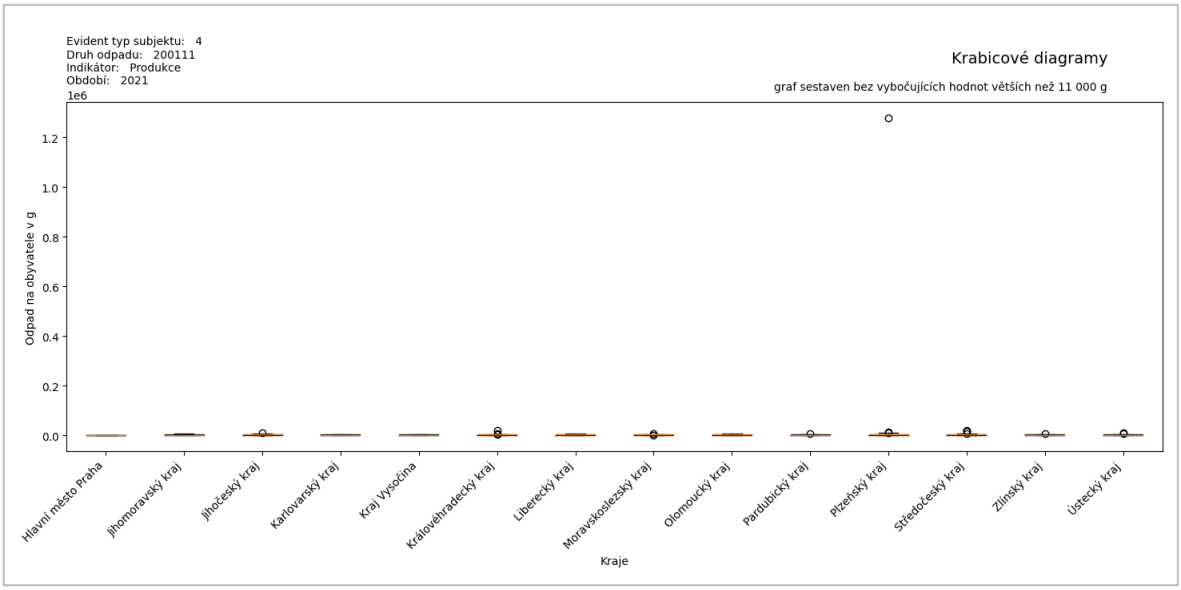
7.5.7 Krabicové diagramy

Jako další nástroj pro získání rychlého přehledu o rozdílech v poloze dat a jejich variabilitě byl zvolen krabicový diagram neboli boxplot. Krabicový diagram je sestaven na datech výběrového souboru, který uživatel zvolil a pohlíží na data na úrovni krajů. Pro každý kraj, který je ve výběrovém souboru, je pro hodnoty množství odpadu na obyvatele v gramech sestaven krabicový diagram. Diagramy za jednotlivé kraje jsou zobrazeny v jednom grafu. Mají společnou osu X a Y. Funkcionalitu řeší v programu funkce nazvaná *boxplot()* a je uvedena v příloze (P XI). Kromě grafu vrací funkce i výstup do textového widgetu v podobě výpisu hodnot, ze kterých je graf sestaven.

7.5.7.1 Popis programového řešení funkce boxplot()

Funkce boxplot() na svém vstupu načítá globální proměnnou vyber_dat_vysledek, která funkci poskytuje výběrový soubor. Pokud není DataFrame prázdná, provádí se výpočet a vložení sloupce ´OdpadNaObyv_g´. Stejně jako při sestavování histogramu (7.5.6.1).

Jednotlivé krabicové diagramy budou zobrazeny v jednom grafu, tak aby se daly kraje mezi sebou porovnávat. Budou proto sdílet společnou osu Y. Pokud bychom použili pro vykreslení krabicových diagramů tuto DataFrame, potýkali bychom se s vybočujícími hodnotami, které by znemožňovaly dostatečně vizualizovat kraje, u kterých se tyto extrémní hodnoty nevyskytují (Obrázek 58). Proto stejně jako u histogramu budou v případě výskytu vybočujících hodnot vybrány pouze ty záznamy, které mají hodnotu ve sloupci ´OdpadNaObyv_g´ menší než je horní hranice zjištěná funkcí *odlehle_hodnoty()*.



Obrázek 58. Krabicový diagram s odlehlými hodnotami

O úpravě vstupního souboru je uživatel informován v popisku grafu, který je umístěn do pravého horního rohu.

Pro vytvoření jednotlivých krabicových diagramů je z vyfiltrované vstupní DataFrame vysledky_lower vytvořen list, který obsahuje unikátní názvy krajů v ní obsažené. Poté se prostřednictvím cyklu for a metody loc[] pro jednotlivé kraje vytvoří samostatné DataFrame se sloupcem ´OdpadNaObyv_g´. Závěrem se pomocí ax.boxplot() vytvoří krabicový diagram pro každou DataFrame kraje a zobrazí se v jednom grafu.

Graf je opatřen popiskem v levém horním rohu, který specifikuje volby uživatele pro sestavení výběrového souboru.

Aby měl uživatel přehled o tom, z jakých dat jsou diagramy sestaveny, zobrazuje se do textového widgetu jednak DataFrame s vybočujícími hodnotami (pokud existuje) a také DataFrame vysledek_lower s daty, ze kterých je graf skutečně vytvořen (Obrázek 59 pozn. názvy ORP a ZÚJ jsou záměrně skryty).

Data pro sestavení krabicových diagramů:

Záznamy (3), které obsahují vybočující hodnoty ve sloupci 'OdpadNaObyv_g':

Kraj_Nazev	ORP_Nazev	ZUJ_Nazev	ZUJ_Cislo	OdpadNaObyv_g	Pocet_Obyvatel	Odpad_vKg
Moravskoslezský kraj				36 494	1 524	55 617
Olomoucký kraj				10 996	3 042	33 452
Zlínský kraj				7 657	140	1 072

Záznamy (506) bez vybočujících hodnot, pro zobrazení v krabicových diagramech:

Kraj_Nazev	ORP_Nazev	ZUJ_Nazev	ZUJ_Cislo	OdpadNaObyv_g	Pocet_Obyvatel	Odpad_vKg
Moravskoslezský kraj	Bohumín			22	20 450	460
Moravskoslezský kraj	Bohumín			96	7 614	737
Moravskoslezský kraj	Bruntál			1	15 523	29
Moravskoslezský kraj	Bruntál			313	411	129
Moravskoslezský kraj	Bruntál			1 526	919	1 403
Moravskoslezský kraj	Bruntál			2 378	529	1 258

Obrázek 59. Textový widget zobrazený funkcí *boxplot()*

Do textového widgetu je zobrazena i DataFrame s charakteristikami poloh odpadu na obyvatele v jednotlivých krajích sestavená nad stejným výběrovým souborem jako krabicový graf čili nad DataFrame očištěnou od vybočujících hodnot (Obrázek 60).

Charakteristiky polohy odpadu na obyvatele v jednotlivých krajích (g) :

Kraj	Pocet_hodnot	Prumer	Median	Minimum	25.percentil	50.percentil	75.percentil	Maximum	Smerodatna_odchylka
Jihomoravský kraj	244	3 856	3 671	27	2 718	3 671	4 990	8 756	1 712
Jihočeský kraj	155	2 441	2 265	0	1 440	2 265	3 202	8 229	1 451
Karlovarský kraj	55	1 938	1 697	189	1 104	1 697	2 639	6 490	1 236
Kraj Vysočina	279	3 068	2 973	11	1 464	2 973	4 464	8 993	2 154
Královéhradecký kraj	150	3 496	3 307	107	2 355	3 307	4 195	9 570	1 722
Liberecký kraj	54	2 297	2 228	367	1 346	2 228	3 173	5 396	1 217
Moravskoslezský kraj	145	2 674	2 650	96	2 060	2 650	3 344	6 138	1 126
Olomoucký kraj	207	3 129	3 113	103	2 173	3 113	3 962	6 705	1 315
Pardubický kraj	157	3 461	3 313	30	2 549	3 313	4 419	7 824	1 502
Plzeňský kraj	110	2 948	2 669	53	1 936	2 669	3 505	8 484	1 612
Středočeský kraj	446	2 900	2 739	5	1 907	2 739	3 656	8 637	1 568
Zlínský kraj	145	2 642	2 614	24	1 956	2 614	3 221	7 657	1 223
Ústecký kraj	93	1 767	1 378	6	959	1 378	2 179	7 647	1 396

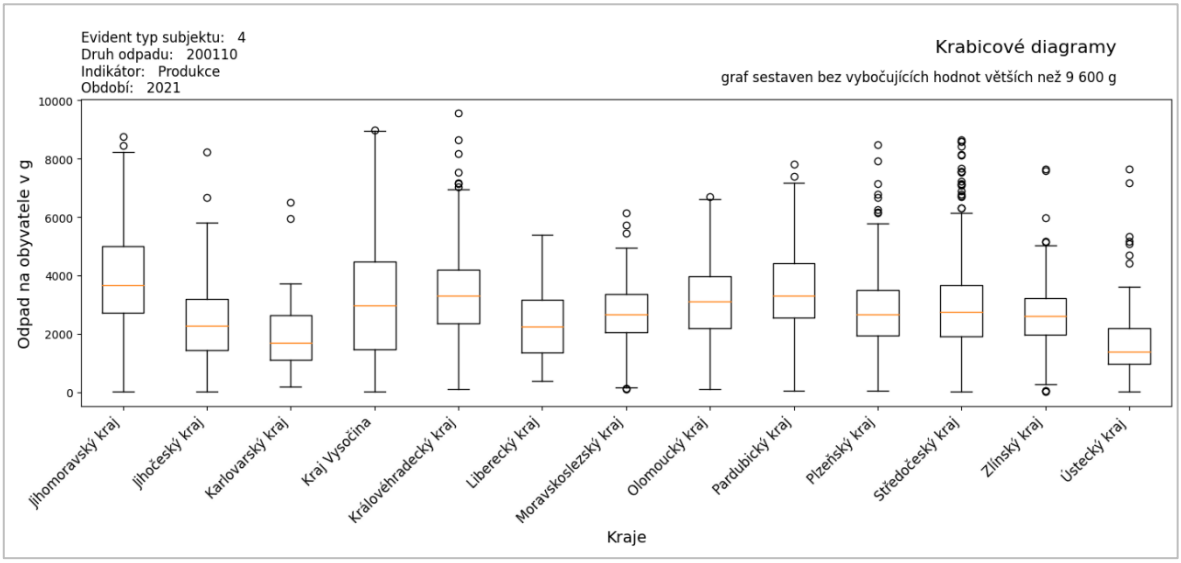
Obrázek 60. Textový widget funkce *boxplot()* uvádějící charakteristiky polohy pro jednotlivé kraje (produkce odpadu 200110 oděvy pro typ subjektu 4)

7.5.7.2 Popis krabicového diagramu jako nástroje pro analýzu dat

Tím, že jsme umístili všechny krabicové diagramy do jednoho grafu se společnými osami X a Y, můžeme odpad v krajích na obyvatele vizuálně velmi dobře vzájemně porovnat (Obrázek 61) hned pomocí pěti číselných charakteristik.

- Medián – Červená čára uvnitř boxu. Jedná se o středovou hodnotu, která odděluje 50 % nižších hodnot od 50 % vyšších hodnot.
- Quartily – Spodní a horní hranice boxu oddělují data na čtvrtiny, tedy na 25 %, 50 % a 75 %. Spodní hranice boxu říká, že 25 % dat je pod touto hranicí. Horní hranice boxu říká, že pod ni spadá 75 % dat.
- IQR – intervalové kvartilové rozpětí tvoří rozdíl mezi třetím a prvním kvartilem

- Fousky – To jsou čáry, které vedou od boxu k extrémním hodnotám v datovém souboru
- Kroužky – Jedná se o individuální hodnoty, které jsou považovány za odlehlé, pokud leží mimo rozsah fousků.



Obrázek 61. Graf s krabicovými diagramy pro produkci oděvního odpadu u obcí

Čím je box větší, tím více dat je v blízkosti mediánu. V grafu se dají jednotlivé regiony porovnávat s ohledem na tyto charakteristiky. Například Středočeský kraj má hodně odlehlých hodnot kdežto Karlovarský kraj vykazuje malý rozptyl, symetrická data a málo odlehlých hodnot.

8 DISKUZE DOSAŽENÝCH VÝSLEDKŮ A NÁVRH DALŠÍHO VÝZKUMU

Existují obecné statistické nástroje, které jsou buďto komerční (např. Statistica, Matlab) nebo volné software např. R, které ale vyžadují hlubší uživatelské dovednosti a orientaci v problematice statistického vyhodnocování dat. Zároveň tyto nástroje nejsou připraveny pro specifické potřeby dat z odpadového hospodářství. To byl důvod, proč vznikla tato bakalářská práce, která si kladla za cíl tuto mezeru vyplnit.

V rámci bakalářské práce vznikl prvotní výpočtový modul, který se snaží na míru problematice odpadového hospodářství data zpracovávat. Jedná se o první verzi, která bude na základě expertního zadání v rámci spolupráce s Vysokým učením technickým v Brně dále rozšiřována v rámci diplomové práce.

V prvotním modulu se podařilo:

- Připravit uživatelské prostředí pro zadání uživatelských voleb a vizualizaci výsledků
- Vytvořit funkcionalitu pro:
    - Načítání dat z CSV souborů a jejich následné uložení do jedné DataFrame.
    - Výběr podmnožiny dat ze vstupní DataFrame, na které budou spouštěny výpočtové funkce.
    - Uložení datasetů do XLSX.
    - Možnost přizpůsobení výstupů do textového widgetu.
    - Aplikaci výpočtových funkcí:
        - Relativní četnosti
        - Histogram četností
        - Krabicový diagram
        - Zjištění odlehlých hodnot
        - Sumarizace dat
        - Seskupení dat
    - Tvorbu map pro geografické úrovně ZÚJ, ORP a krajů.

Zároveň byly objeveny omezení a limity zvoleného přístupu ve formě:

- Funkcionality nejsou úplně obecné a nedají se křížit mezi sebou dle potřeby

- Chybějící statistické vyhodnocení např. korelační analýzy (např. analýza prostorové autokorelace Moran I.) a podobně.
- Omezení mapových výstupů s ohledem na použitá podkladová data.
- Neřeší se běžně používané indikátory odpadového hospodářství (míra separace v obci, materiálové využití odpadu atp.).
- Zobrazení a analýzy časových řad.
- Pro snadnější porovnání grafů by se využila možnost správy os. Nyní je měřítko os vytvářeno automaticky.
- Možnost volby zobrazení a analýz pro celý nebo zvolený datový soubor (např. vybočující hodnoty).
- Proces načítání a kontroly vstupních dat zapracovat do GUI.

Všechny tyto body jsou nad rámec rozsahu bakalářské práce, a proto nebyly řešeny. Cílem však je zapracovat je v rámci diplomové práce.

ZÁVĚR

V rámci bakalářské práce byl vytvořen výpočtový modul pro statistické zpracování dat z odpadového hospodářství. Data v podobě ročního hlášení o produkci a nakládání s odpady byla poskytnuta v rámci řešení projektu CEVOOH řešeného na Ústavu procesního inženýrství Vysokého učení technického v Brně. Cílem bylo připravit univerzální nástroj, kterým by bylo umožněno nezávisle vyhodnocovat různé druhy odpadů za různá období.

Teoretická část práce poskytla prostor pro studium a shrnutí nástrojů popisné statistiky, které byly v praktické části práce využity při vytváření výpočtových funkcí a grafů. Celé řešení bylo naprogramováno pomocí jazyku Python a jeho knihoven Pandas, Matplotlib, GeoPandas a Pyplot, které jsou stručně zmíněny. Výpočtový modul je navržen s respektem obecného postupu pro analýzu dat. Tento postup je představen v další kapitole práce. Teoretická část je uzavřena seznámením čtenáře s odpadovým hospodářstvím. Je vysvětlujícím úvodem pro kapitolu popisující zdrojová data, jež je součástí praktické části práce.

Praktická část nejprve představuje návrh celého řešení výpočtového modulu, poté se věnuje čtyřem krokům: Načtení a kontrole dat, Tvorbě grafického uživatelského rozhraní (GUI), Vytvoření výběrové funkce a Vytvářením funkcionalit pro statistické zpracování dat.

Zmiňované kroky jsou velmi podrobně v práci popsány, včetně ukázek důležitých částí kódů. Uživatelské rozhraní bylo vytvářeno pomocí knihovny Tkinter v Pythonu a poskytuje uživateli možnost zadání specifikací pro získání podmnožiny dat, nad kterými se následně realizují výpočtové funkce a tvoří grafy. Textové výstupy funkcí jsou zobrazovány v textovém widgetu GUI a grafy jsou otevírány v samostatných oknech. Uživateli jsou dány možnosti uložení výstupů do XLSX souboru a přizpůsobení zobrazení výstupů do textového widgetu.

Pro statistické zhodnocení dat byly sestaveny výpočtové funkce řešící zejména oblast popisné statistiky a charakteristiky polohy dat výběrového souboru: Relativní četnosti, Histogram četností, Krabicový diagram, Sumarizace dat a Seskupování dat aj. Velkým problémem u zmíněných funkcionalit byla existence extrémních hodnot ve zdrojových datech, která způsobovala nečitelnost grafů. Z toho důvodu byla výpočtově řešena i funkcionalita Zjištění odlehlých hodnot a o extrémní hodnoty byla upravována i vstupní data při výpočtech. Významným přínosem práce je možnost zobrazovat data v mapách na geografických

úrovních krajů, obcí s rozšířenou působností (ORP) a základních územních jednotek (ZÚJ).

Při kritickém pohledu na realizované řešení byly nalezeny omezení a limity, které jsou shrnuty v poslední části práce. Vzhledem ke své povaze jsou však již nad rámec rozsahu bakalářské práce a budou sloužit jako podklad pro další výzkum např. v rámci diplomové práce.

SEZNAM POUŽITÉ LITERATURY

[1] SEGER, Jan a Richard HINDLS. *Statistické metody v tržním hospodářství*. Praha: Victoria Publishing, 1995, 435 s. ISBN 8071870587.

[2] BUDÍKOVÁ, Marie, Maria KRÁLOVÁ a Bohumil MAROŠ. *Průvodce základními statistickými metodami*. Praha: Grada, 2010, 272 s. Expert. ISBN 9788024732435.

[3] NEUBAUER, Jiří, Marek SEDLAČÍK a Oldřich KŘÍŽ. *Základy statistiky: aplikace v technických a ekonomických oborech*. 2., rozšířené vydání. Praha: Grada, 2016, 278 s. ISBN 9788024757865.

[4] ANDĚL, Jiří. *Základy matematické statistiky*. 2., opr. vyd. Praha: Matfyzpress, 2007, 358 s. ISBN 9788073780012.

[5] GIBILISCO, Stan. *Statistika bez předchozích znalostí*. Brno: Computer Press, 2009, 272 s. ISBN 9788025124659.

[6] MAREK, Luboš. *Statistika v příkladech*. Praha: Professional Publishing, 2013, 403 s. ISBN 9788074311185.

[7] NOVÁKOVÁ, Kateřina a Petr VESELÝ. *Jazyk R a tvorba grafů*. Praha: Grada Publishing, 2021, 253 s. Knihovna programátora. ISBN 978-80-271-3137-2.

[8] JARKOVSKÝ, Jiří. Analýza a management dat pro zdravotnické obory, Analýza klinických dat: Identifikace odlehlých hodnot. *Matematická biologie* [online]. Institut biostatistiky a analýz Lékařské fakulty Masarykovy univerzity [cit. 2023-05-14]. Dostupné z: https://portal.matematickabiologie.cz/index.php?pg=aplikovana-analyza-klinickych-a-biologickych-dat--biostatistika-pro-matematickou-biologii--data-jejich-popis-a-vizualizace--identifikace-odlehlych-hodnot#priklad%203.5

[9] ŠTĚPÁNEK, Jindřich. Histogram představuje grafické zobrazení intervalového zobrazení četnosti znaku jakosti slouží k názornému zobrazení „struktury“ naměřených dat hranice. In: *SlidePlayer* [online]. 2021 [cit. 2023-05-13]. Dostupné z: https://slideplayer.cz/slide/1899225/

[10] HOŠEK, Petr. Statistika. *Po studium* [online]. Plzeň: Lékařská fakulta UK v Plzni, 2015 [cit. 2023-05-14]. Dostupné z: https://postudium.cz/pluginfile.php/7248/mod_resource/content/6/statistika-html.html#_idTextAnchor023

[11] ROHLA, Jonáš. Box – plot OA a VOŠ Příbram. In: *SlidePlayer* [online]. 2021 [cit. 2023-05-13]. Dostupné z: https://slideplayer.cz/slide/2734155/

[12] LITSCHMANNOVÁ, Martina a Adéla VRTKOVÁ. Úvod do teorie odhadu. *VSB Technická univerzita Ostrava* [online]. Ostrava [cit. 2023-05-14]. Dostupné z: http://am-nas.vsb.cz/lit40/STATLAB/ZS_neurcitost.pdf

[13] HOLEC, Matěj. Chybějící a odlehlé hodnoty: Odstranění odlehlých hodnot pomocí algoritmu k-means. *Dokuwiki* [online]. ČVUT FEL, 2011 [cit. 2023-05-14]. Dostupné z: https://cw.fel.cvut.cz/old/_media/courses/m33sad/2_cviceni.pdf

[14] CARBONNELLE, Pierre. PYPL PopularitY of Programming Language. *Pypl* [online]. 2023 [cit. 2023-05-14]. Dostupné z: https://pypl.github.io/PYPL.html

[15] Top programming languages for data scientists in 2023. *DataCamp* [online]. 2023 [cit. 2023-05-14]. Dostupné z: https://www.datacamp.com/blog/top-programming-languages-for-data-scientists-in-2022

[16] WALKER, Michael. *Python Data Cleaning Cookbook: Modern techniques and Python tools to detect and remove dirty data and extract key insights*. Birmingham: Packt Publishing, 2020. ISBN ISBN 978-1-80056-566-1.

[17] COTTON, Richie. Data manipulation with Pandas. *Datacamp* [online]. 2022 [cit. 2023-05-14]. Dostupné z: https://campus.datacamp.com/courses/data-manipulation-with-pandas/transforming-dataframes?ex=4

[18] CASE, Erin. Introduction to Seaborn. *Datacamp* [online]. 2022 [cit. 2023-05-14]. Dostupné z: https://campus.datacamp.com/courses/data-manipulation-with-pandas/transforming-dataframes?ex=4

[19] Python pro data 1: Programování hravě i vážně. *Kódím* [online]. [cit. 2023-03-04]. Dostupné z: https://kodim.cz/kurzy/python-data-1/python-pro-data-1/agregace-a-spojovani/pokrocile-upravy

[20] Pandas. *Pandas* [online]. Pydata, c2023 [cit. 2023-03-27]. Dostupné z: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html

[21] Pandas Series. *W3schools* [online]. [cit. 2023-05-13]. Dostupné z: https://www.w3schools.com/python/pandas/pandas_series.asp

[22] VALKENBURG, Mary Van. DATACAMP. *GeoJSON: Visualizing geospatial data in Python* [online]. Datacamp, 2022 [cit. 2023-05-13]. Dostupné z:

https://campus.datacamp.com/courses/visualizing-geospatial-data-in-python/creating-and-joining-geodataframes?ex=8

[23] *GeoJSON* [online]. [cit. 2023-05-01]. Dostupné z: https://geojson.org/

[24] Introduction to GUI With Tkinter in Python. *DataCamp* [online]. Dec 2019 [cit. 2023-05-15]. Dostupné z: http://tkinter.programujte.com/

[25] Úvod do Tkinter. *Programujte.com* [online]. [cit. 2023-05-15]. Dostupné z: http://tkinter.programujte.com/

[26] FRIEDEN, B. Roy a Robert A. GATENBY, ed. *Exploratory data analysis using Fisher information*. London: Springer, 2007, xiii, 363 s. Dostupné z: doi:9781846287770

[27] CALZON, Bernardita. Your Modern Business Guide To Data Analysis Methods And Techniques. *Datapine: The datapine Blog* [online]. Berlin, Mar 3rd 2023 [cit. 2023-05-17]. Dostupné z: https://www.datapine.com/blog/data-analysis-methods-and-techniques/#data-analysis-definition

[28] MINISTERSTVO ŽIVOTNÍHO PROSTŘEDÍ. Plán odpadového hospodářství České republiky na období 2015–2024. *Mpz* [online]. Praha, květen 2014 [cit. 2023-03-10]. Dostupné z: https://www.mzp.cz/C1257458002F0DC7/cz/news_140506_Plan_odpady/$FILE/Plan_odpadoveho_hospodarstvi-060514.pdf

[29] ČESKÁ REPUBLIKA. Zákon č. 541/2020 Sb.: Zákon o odpadech. In: *Sbírka zákonů*. 2020, ročník 2020, 222/2020, číslo 541.

[30] ČESKÁ REPUBLIKA. Zákon č. 273/2021 Sb.: Vyhláška o podrobnostech nakládání s odpady. In: *Sbírka zákonů*. 2021, ročník 2021, 119/2021, číslo 273.

[31] Příloha č.20 - Hlášení o produkci a nakládání s odpady. *Praha 5* [online]. 19.11.2018 [cit. 2023-02-25]. Dostupné z: https://www.praha5.cz/hlaseni-o-produkci-a-nakladani-s-odpady/

[32] ŠKOP, Michal. *Kartogram ČR (choropleth, choropletová mapa)* [online]. PORTÁL OTEVŘENÝCH DAT. 2020, 23. 4. 2020 [cit. 2023-02-22]. Dostupné z: https://data.gov.cz/%C4%8Dl%C3%A1nky/kartogram-choropleth

[33] HALÁSEK, Jiří. Malý lexikon obcí České republiky - 2022. *Český statistický úřad* [online]. 2022, 15. prosince 2022 [cit. 2023-04-05]. Dostupné z: https://www.czso.cz/csu/czso/maly-lexikon-obci-ceske-republiky-2022

[34] WENISCH, Martin. České obce. *Github* [online]. 2022, 15. listopadu 2022 [cit. 2023-04-10]. Dostupné z: https://github.com/cesko-digital/obce

[35] HAVEL, Marek. Počet obyvatel v obcích – k 1.1.2022. *Český statistický úřad* [online]. 2022, 29. dubna 2022 [cit. 2023-03-18]. Dostupné z: https://www.czso.cz/csu/czso/pocet-obyvatel-v-obcich-k-112022

[36] *The Matplotlib: Choosing Colormaps in Matplotlib* [online]. 2021–2023 [cit. 2023-05-01]. Dostupné z: https://matplotlib.org/stable/tutorials/colors/ color-maps.html

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

GUI	Grafické uživatelské rozhraní
ISOH	Informační systém odpadového hospodářství.
JSON	JavaScript Object Notation
ORP	Obce s rozšířenou působností
ZÚJ	Základní územní jednotka.

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PROGRAMŮ

SEZNAM PŘÍLOH

PŘÍLOHA P I: KATALOG ODPADŮ – SKUPINA 20

20	**KOMUNÁLNÍ ODPADY (ODPADY Z DOMÁCNOSTÍ A PODOBNÉ ŽIVNOSTENSKÉ, PRŮMYSLOVÉ ODPADY A ODPADY Z ÚŘADŮ), VČETNĚ SLOŽEK Z ODDĚLENÉHO SBĚRU**
20 01	**Složky z odděleného sběru (kromě odpadů uvedených v podskupině 15 01)**
20 01 01	Papír a lepenka
20 01 02	Sklo
20 01 08	Biologicky rozložitelný odpad z kuchyní a stravoven
20 01 10	Oděvy
20 01 11	Textilní materiály
20 01 13*	Rozpouštědla
20 01 14*	Kyseliny
20 01 15*	Zásady
20 01 17*	Fotochemikálie
20 01 19*	Pesticidy
20 01 21*	Zářivky a jiný odpad obsahující rtuť
20 01 23*	Vyřazená zařízení obsahující chlorofluorouhlovodíky
20 01 25	Jedlý olej a tuk
20 01 26*	Olej a tuk neuvedený pod číslem 20 01 25
20 01 27*	Barvy, tiskařské barvy, lepidla a pryskyřice obsahující nebezpečné látky
20 01 28	Barvy, tiskařské barvy, lepidla a pryskyřice neuvedené pod číslem 20 01 27
20 01 29*	Detergenty obsahující nebezpečné látky
20 01 30	Detergenty neuvedené pod číslem 20 01 29
20 01 31*	Nepoužitelná cytostatika
20 01 32*	Jiná nepoužitelná léčiva neuvedená pod číslem 20 01 31
20 01 33*	Baterie a akumulátory, zařazené pod čísly 16 06 01, 16 06 02 nebo pod číslem 16 06 03 a netříděné baterie a akumulátory obsahující tyto baterie
20 01 34	Baterie a akumulátory neuvedené pod číslem 20 01 33
20 01 35*	Vyřazené elektrické a elektronické zařízení obsahující nebezpečné látky neuvedené pod čísly 20 01 21 a 20 01 23 6)
20 01 36	Vyřazené elektrické a elektronické zařízení neuvedené pod čísly 20 01 21, 20 01 23 a 20 01 35
20 01 37*	Dřevo obsahující nebezpečné látky
20 01 38	Dřevo neuvedené pod číslem 20 01 37
20 01 39	Plasty
20 01 40	Kovy
20 01 41	Odpady z čištění komínů
20 01 99	Další frakce jinak blíže neurčené
20 02	**Odpady ze zahrad a parků (včetně hřbitovního odpadu)**
20 02 01	Biologicky rozložitelný odpad
20 02 02	Zemina a kameny
20 02 03	Jiný biologicky nerozložitelný odpad
20 03	**Ostatní komunální odpady**
20 03 01	Směsný komunální odpad
20 03 02	Odpad z tržišť
20 03 03	Uliční smetky
20 03 04	Kal ze septiků a žump
20 03 06	Odpad z čištění kanalizace
20 03 07	Objemný odpad
20 03 99	Komunální odpady jinak blíže neurčené

PŘÍLOHA P II: KÓDOVÁNÍ ZPŮSOBŮ NAKLÁDÁNÍ S ODPADY

KÓDOVÁNÍ ZPŮSOBŮ NAKLÁDÁNÍ S ODPADY			
PŮVOD ODPADŮ			
Původ odpadů	**Kód**	**Množství odpadu +/-**	**Partner**
Produkce odpadu (vlastní vyprodukovaný odpad)	A00	(+)	NE
Odpad převzatý od původce, jiné oprávněné osoby (sběr, výkup, shromažďování), nebo jiné provozovny	B00	(+)	ANO
Množství odpadu převedené z minulého roku (zůstatek na skladu k 1. lednu vykazovaného roku)	C00	(+)	NE
ZPŮSOB NAKLÁDÁNÍ S ODPADY			
Využívání odpadů			
Způsob	**kód**	**+/-**	**partner**
Využití odpadu způsobem obdobným jako paliva nebo jiným způsobem k výrobě energie	XR1	(-)	NE
Získání /regenerace rozpouštědel	XR2	(-)	NE
Získání/regenerace organických látek, které se nepoužívají jako rozpouštědla (včetně biologických procesů mimo kompostování a biologickou dekontaminaci)	XR3	(-)	NE
Recyklace/znovuzískání kovů a kovových sloučenin	XR4	(-)	NE
Recyklace/znovuzískání ostatních anorganických materiálů	XR5	(-)	NE
Regenerace kyselin a zásad	XR6	(-)	NE
Obnova látek používaných ke snižování znečištění	XR7	(-)	NE
Získání složek katalyzátorů	XR8	(-)	NE
Rafinace použitých olejů nebo jiný způsob opětného použití olejů	XR9	(-)	NE
Aplikace do půdy, která je přínosem pro zemědělství nebo zlepšuje ekologii	XR10	(-)	NE
Využití odpadů, které vznikly aplikací některého z postupů uvedených pod označením R1 až R10	XR11	(-)	NE
Předúprava odpadů k aplikaci některého z postupů uvedených pod označením R1 až R11	XR12	(-)	NE
Skladování materiálů před aplikací některého z postupů uvedených pod označením R1 až R12 (s výjimkou dočasného skladování na místě vzniku před sběrem) k 31. prosinci vykazovaného roku	XR13	(-)	NE
Odstraňování odpadů			
Způsob	**kód**	**+/-**	**partner**
Ukládání v úrovni nebo pod úrovní terénu (skládkování)	XD1	(-)	NE
Úprava půdními procesy (např. biologický rozklad kapalných odpadů či kalů v půdě, apod.)	XD2	(-)	NE
Hlubinná injektáž (např. injektáž čerpatelných kapalných odpadů do vrtů, solných komor nebo prostor přírodního původu, apod.)	XD3	(-)	NE
Ukládání do povrchových nádrží (např. vypouštění kapalných odpadů nebo kalů do prohlubní, vodních nádrží, lagun, apod.)	XD4	(-)	NE
Ukládání do speciálně technicky provedených skládek (např. ukládání do oddělených, utěsněných, zavřených prostor izolovaných navzájem i od okolního prostředí, apod.)	XD5	(-)	NE
Biologická úprava jinde v této příloze nespecifikovaná, jejímž konečným produktem jsou sloučeniny nebo směsi, které se odstraňují některým z postupů uvedených pod označením Dl až D12	XD8	(-)	NE
Fyzikálně-chemická úprava jinde v této příloze nespecifikovaná, jejímž konečným produktem jsou sloučeniny nebo směsi, které se odstraňují některým z postupů uvedených pod označením D1 až D12 (např. odpařování, sušení, kalcinace)	XD9	(-)	NE
Spalování na pevnině	XD10	(-)	NE
Konečné či trvalé uložení (např. ukládání v kontejnerech do dolů)	XD12	(-)	NE
Úprava složení nebo smíšení odpadů před jejich odstraněním některým z postupů uvedených pod označením D1 až D12	XD13	(-)	NE
Úprava jiných vlastností odpadů (kromě úpravy zahrnuté do D13) před jejich odstraněním některým z postupů uvedených pod označením D1 až D13	XD14	(-)	NE
Skladování materiálů před jejich odstraněním některým z postupů uvedených pod označením Dl až D14 (s výjimkou dočasného skladování na místě vzniku před shromážděním potřebného množství) k 31. prosinci vykazovaného roku	XD15	(-)	NE

Ostatní			
Způsob	**kód**	**+/-**	**partner**
Využití odpadů s výjimkou využívání kalů podle vyhl. 382/2001 Sb. na terénní úpravy apod.	XN1	(-)	NE
Předání kalů ČOV k použití na zemědělské půdě	XN2	(-)	ANO
Předání jiné oprávněné osobě (kromě přepravce, dopravce), nebo jiné provozovně	XN3	(-)	ANO
Zůstatek na skladu k 31. prosinci vykazovaného roku	XN5	(-)	NE
Přeshraniční přeprava odpadu z členského státu EU do CR	BN6	(+)	ANO
Přeshraniční přeprava odpadu do členského státu EU z ČR	XN7	(-)	ANO
Předání (dílů, odpadů) pro opětovné použití	XN8	(-)	ANO
Zpracování autovraku	XN9	(-)	NE
Prodej odpadu jako suroviny („druhotné suroviny")	XN10	(-)	ANO
Využití odpadu na rekultivace skládek	XN11	(-)	NE
Ukládání odpadů jako technologický materiál na zajištění skládky	XN12	(-)	NE
Kompostování	XN13	(-)	NE
Biologická dekontaminace	XN14	(-)	NE
Protektorování pneumatik	XN15	(-)	NE
Dovoz odpadu ze státu, který není členským státem EU	BN16	(+)	ANO
Vývoz odpadu do státu, který není členským státem EU	XN17	(-)	ANO
Zpracování elektroodpadu	XN18	(-)	NE
Převzetí zpětně odebraných některých výrobků nebo zpětně odebraných elektrozařízení od právnické osoby nebo fyzické osoby oprávněné k podnikání, která zajišťuje zpětný odběr podle § 37k nebo § 38 zákona nebo převzetí odpadů od nepodnikajících fyzických osob - občanů	BN30	(+)	ANO
Odpad po úpravě, když nedošlo ke změně katalogového čísla odpadu	BN40	(+)	NE
Inventurní rozdíl - vyrovnání nedostatku odpadu	XN50	(+)	NE
Inventurní rozdíl - vyrovnání přebytku odpadu	XN53	(-)	NE
Staré zátěže, živelní pohromy, černé skládky apod.	XN60	(+)	NE
Staré zátěže, živelní pohromy, černé skládky apod.	XN63	(-)	NE

PŘÍLOHA III: UPRAVENÁ TABULKA KODOVANI ZPŮSOBŮ NAKLÁDÁNÍ S ODPADY

Kod	Operator	Uvedeni_Partnera	Navyseni_Ubytek	Kodovani_zpusobu	Puvod_Zpusob	Popis_zpusobu
A00	(+)	NE	1	Původ odpadů	Původ odpadů	Produkce odpadu (vlastní vyprodukovaný odpad)
B00	(+)	ANO	1	Původ odpadů	Původ odpadů	Odpad převzatý od původce, jiné oprávněné osoby (sběr, výkup,
BN16	(+)	ANO	1	Způsob nakládání s odpady	Ostatní	Dovoz odpadu ze státu, který není členským státem EU
BN30	(+)	ANO	1	Způsob nakládání s odpady	Ostatní	Převzetí zpětně odebraných některých výrobků nebo zpětně
BN40	(+)	NE	1	Způsob nakládání s odpady	Ostatní	Odpad po úpravě, když nedošlo ke změně katalogového čísla
BN6	(+)	ANO	1	Způsob nakládání s odpady	Ostatní	Přeshraniční přeprava odpadu z členského státu EU do CR
C00	(+)	NE	1	Původ odpadů	Původ odpadů	Množství odpadu převedené z minulého roku (zůstatek na skladu
Předání	(-)	ANO	-1			
Převzetí	(+)	ANO	1			
XD1	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Ukládání v úrovni nebo pod úrovní terénu (skládkování)
XD10	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Spalování na pevnině
XD12	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Konečné či trvalé uložení (např. ukládání v kontejnerech do dolů)
XD13	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Úprava složení nebo smíšení odpadů před jejich odstraněním
XD14	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Úprava jiných vlastností odpadů (kromě úpravy zahrnuté do D13)
XD15	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Skladování materiálů před jejich odstraněním některým z postupů
XD2	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Úprava půdními procesy (např. biologický rozklad kapalných
XD3	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Hlubinná injektáž (např. injektáž čerpatelných kapalných odpadů
XD4	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Ukládání do povrchových nádrží (např. vypouštění kapalných
XD5	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Ukládání do speciálně technicky provedených skládek (např.
XD8	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Biologická úprava jinde v této příloze nespecifikovaná, jejímž
XD9	(-)	NE	-1	Způsob nakládání s odpady	Odstraňování odpadů	Fyzikálně-chemická úprava jinde v této příloze nespecifikovaná,
XN1	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Využití odpadů s výjimkou využívání kalů podle vyhl. 382/2001
XN10	(-)	ANO	-1	Způsob nakládání s odpady	Ostatní	Prodej odpadu jako suroviny („druhotné suroviny")
XN11	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Využití odpadu na rekultivace skládek
XN12	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Ukládání odpadů jako technologický materiál na zajištění skládky
XN13	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Kompostování
XN14	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Biologická dekontaminace
XN15	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Protektorování pneumatik
XN17	(-)	ANO	-1	Způsob nakládání s odpady	Ostatní	Vývoz odpadu do státu, který není členským státem EU
XN18	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Zpracování elektroodpadu
XN2	(-)	ANO	-1	Způsob nakládání s odpady	Ostatní	Předání kalů ČOV k použití na zemědělské půdě
XN3	(-)	ANO	-1	Způsob nakládání s odpady	Ostatní	Předání jiné oprávněné osobě (kromě přepravce, dopravce), nebo
XN5	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Zůstatek na skladu k 31. prosinci vykazovaného roku
XN50	(+)	NE	1	Způsob nakládání s odpady	Ostatní	Inventurní rozdíl - vyrovnání nedostatku odpadu
XN53	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Inventurní rozdíl - vyrovnání přebytku odpadu
XN60	(+)	NE	1	Způsob nakládání s odpady	Ostatní	Staré zátěže, živelní pohromy, černé skládky apod.
XN63	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Staré zátěže, živelní pohromy, černé skládky apod.
XN7	(-)	ANO	-1	Způsob nakládání s odpady	Ostatní	Přeshraniční přeprava odpadu do členského státu EU z ČR
XN8	(-)	ANO	-1	Způsob nakládání s odpady	Ostatní	Předání (dílů, odpadů) pro opětovné použití
XN9	(-)	NE	-1	Způsob nakládání s odpady	Ostatní	Zpracování autovraku
XR1	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Využití odpadu způsobem obdobným jako paliva nebo jiným
XR10	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Aplikace do půdy, která je přínosem pro zemědělství nebo zlepšuje
XR11	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Využití odpadů, které vznikly aplikací některého z postupů
XR12	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Předúprava odpadů k aplikaci některého z postupů uvedených
XR13	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Skladování materiálů před aplikací některého z postupů uvedených
XR2	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Získání /regenerace rozpouštědel
XR3	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Získání/regenerace organických látek, které se nepoužívají jako
XR4	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Recyklace/znovuzískání kovů a kovových sloučenin
XR5	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Recyklace/znovuzískání ostatních anorganických materiálů
XR6	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Regenerace kyselin a zásad
XR7	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Obnova látek používaných ke snižování znečištění
XR8	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Získání složek katalyzátorů
XR9	(-)	NE	-1	Způsob nakládání s odpady	Využívání odpadů	Rafinace použitých olejů nebo jiný způsob opětného použití olejů

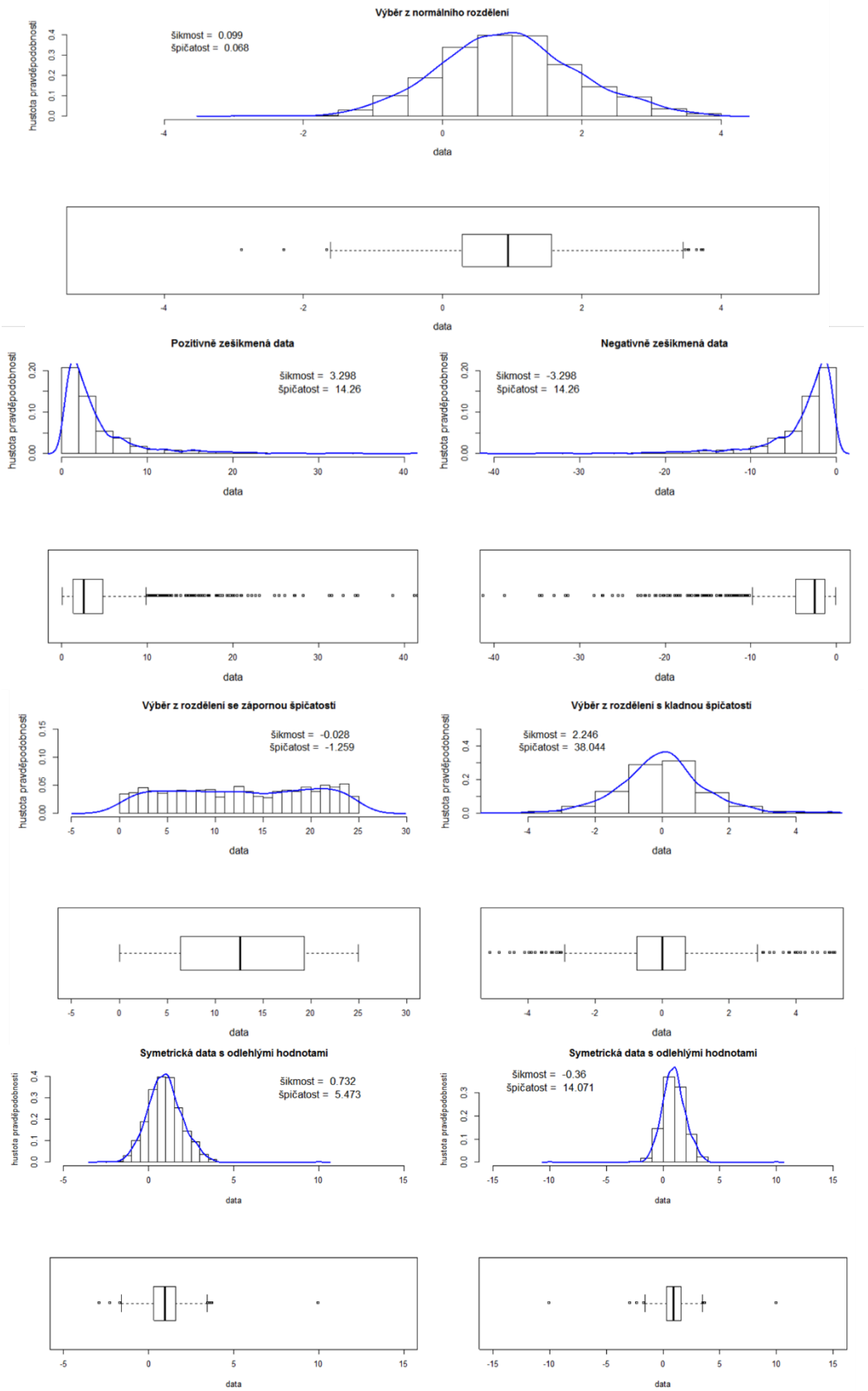
PŘÍLOHA IV: AGREGAČNÍ TABULKA ZÚJ-ORP-KRAJ

ZUJ _Kod	ZUJ_Nazev	ORP_Kod	ORP_Nazev	Kraj_Kod	Kraj_Nazev
500011	Želechovice nad Dřevnicí	7213	Zlín	CZ072	Zlínský
500020	Petrov nad Desnou	7111	Šumperk	CZ071	Olomoucký
500046	Libhošť	8115	Nový Jičín	CZ080	Moravskoslezský
500062	Krhová	7210	Valašské Meziříčí	CZ072	Zlínský
500071	Poličná	7210	Valašské Meziříčí	CZ072	Zlínský
500101	Bražec	4103	Karlovy Vary	CZ041	Karlovarský
500127	Doupovské Hradiště	4106	Ostrov	CZ041	Karlovarský
500135	Kozlov	7107	Olomouc	CZ071	Olomoucký
500151	Luboměř pod Strážnou	7101	Hranice	CZ071	Olomoucký
500160	Město Libavá	7110	Šternberk	CZ071	Olomoucký
500194	Polná na Šumavě	3103	Český Krumlov	CZ031	Jihočeský
500259	Veřovice	8105	Frenštát pod Radhoštěm	CZ080	Moravskoslezský
500291	Vřesina	8119	Ostrava	CZ080	Moravskoslezský
500496	Olomouc	7107	Olomouc	CZ071	Olomoucký
500526	Bělkovice-Lašťany	7107	Olomouc	CZ071	Olomoucký
500623	Bílá Lhota	7105	Litovel	CZ071	Olomoucký
500801	Blatec	7107	Olomouc	CZ071	Olomoucký
500852	Bohuňovice	7107	Olomouc	CZ071	Olomoucký
500861	Bouzov	7105	Litovel	CZ071	Olomoucký
500879	Bystročice	7107	Olomouc	CZ071	Olomoucký
501476	Dlouhá Loučka	7112	Uničov	CZ071	Olomoucký
501646	Dolany	7107	Olomouc	CZ071	Olomoucký
501751	Drahanovice	7107	Olomouc	CZ071	Olomoucký
501794	Dub nad Moravou	7107	Olomouc	CZ071	Olomoucký
501841	Grygov	7107	Olomouc	CZ071	Olomoucký
502146	Hlubočky	7107	Olomouc	CZ071	Olomoucký
502235	Hněvotín	7107	Olomouc	CZ071	Olomoucký
502405	Hnojice	7110	Šternberk	CZ071	Olomoucký
502545	Horka nad Moravou	7107	Olomouc	CZ071	Olomoucký
502839	Cholina	7105	Litovel	CZ071	Olomoucký
503142	Jívová	7110	Šternberk	CZ071	Olomoucký
503304	Kožušany-Tážaly	7107	Olomouc	CZ071	Olomoucký
503410	Žitovlice	2118	Nymburk	CZ020	Středočeský
503444	Litovel	7105	Litovel	CZ071	Olomoucký
503622	Luká	7105	Litovel	CZ071	Olomoucký
503657	Lutín	7107	Olomouc	CZ071	Olomoucký
503738	Majetín	7107	Olomouc	CZ071	Olomoucký
503916	Kostelec	3213	Stříbro	CZ032	Plzeňský
503941	Libavá	7107	Olomouc	CZ071	Olomoucký
504246	Mladeč	7105	Litovel	CZ071	Olomoucký
504301	Dřenice	5304	Chrudim	CZ053	Pardubický
504441	Náklo	7105	Litovel	CZ071	Olomoucký
504505	Náměšť na Hané	7107	Olomouc	CZ071	Olomoucký

PŘÍLOHA V: POČET OBYVATEL V OBCÍCH ČESKÉ REPUBLIKY

Počet obyvatel v obcích České republiky k 1. 1. 2021								
Population of municipalities of the Czech Republic, 1 January 2021								
Kód *Code of*		Název obce	Počet obyvatel *Population*			Průměrný věk *Average age*		
okresu	obce		celkem	muži	ženy	celkem	muži	ženy
district	*municipality*	*Name of municipality*	*Total*	*Men*	*Women*	*Total*	*Men*	*Women*
CZ0100	554782	Praha	1259413	613316	646097	42,4	40,8	43,8
Okres Benešov								
CZ0201	529303	Benešov	16525	7880	8645	43,3	41,6	44,8
CZ0201	532568	Bernartice	227	107	120	44,9	48,1	42,0
CZ0201	530743	Bílkovice	215	105	110	43,7	44,1	43,3
CZ0201	532380	Blažejovice	101	49	52	48,0	46,2	49,8
CZ0201	532096	Borovnice	82	40	42	43,8	41,9	45,6
CZ0201	532924	Bukovany	762	388	374	41,0	41,3	40,7
CZ0201	529451	Bystřice	4433	2213	2220	41,9	40,9	42,9
CZ0201	532690	Ctiboř	152	80	72	36,4	36,9	35,9
CZ0201	529478	Čakov	129	68	61	39,0	37,3	41,0
CZ0201	529486	Čechtice	1390	686	704	43,4	41,8	44,9
CZ0201	529516	Čerčany	2874	1426	1448	41,9	41,4	42,4
CZ0201	529532	Červený Újezd	346	172	174	41,8	41,9	41,8
CZ0201	529541	Český Šternberk	171	85	86	42,9	43,7	42,2
CZ0201	529567	Čtyřkoly	714	346	368	41,4	40,8	41,9
CZ0201	532746	Děkanovice	60	31	29	43,2	44,1	42,2
CZ0201	529621	Divišov	1747	874	873	40,9	39,3	42,6
CZ0201	529648	Dolní Kralovice	877	438	439	44,3	42,4	46,2
CZ0201	532151	Drahňovice	99	52	47	37,2	33,8	41,0
CZ0201	532843	Dunice	67	37	30	43,8	41,0	47,2
CZ0201	529702	Heřmaničky	717	357	360	43,3	43,1	43,5
CZ0201	532932	Hradiště	28	19	9	50,1	45,8	59,3
CZ0201	529737	Hulice	276	135	141	44,9	44,2	45,5
CZ0201	529745	Hvězdonice	330	167	163	42,0	41,2	42,7
CZ0201	532886	Chářovice	212	109	103	37,0	36,5	37,5
CZ0201	532878	Chleby	45	26	19	51,1	49,3	53,7
CZ0201	532045	Chlístov	402	206	196	39,5	39,2	39,7
CZ0201	529770	Chlum	122	65	57	46,1	46,5	45,6
CZ0201	529788	Chmelná	149	82	67	39,6	37,3	42,6
CZ0201	529796	Chocerady	1255	633	622	41,3	40,9	41,8
CZ0201	532606	Choratice	70	36	34	45,2	41,7	49,0
CZ0201	529818	Chotýšany	589	289	300	39,8	39,1	40,5
CZ0201	532037	Chrášťany	232	115	117	42,1	40,9	43,4
CZ0201	529842	Jankov	897	420	477	43,4	42,0	44,6
CZ0201	529851	Javorník	122	63	59	45,4	44,8	46,0
CZ0201	532134	Ješetice	128	64	64	42,2	40,1	44,3
CZ0201	531031	Kamberk	142	76	66	48,7	47,1	50,6
CZ0201	529907	Keblov	183	97	86	44,4	42,5	46,4
CZ0201	533084	Kladruby	270	144	126	42,4	40,5	44,5
CZ0201	529931	Kondrac	518	252	266	40,0	40,8	39,3
CZ0201	529940	Kozmice	346	159	187	39,5	38,6	40,3
CZ0201	529958	Krhanice	1027	519	508	41,2	40,1	42,3
CZ0201	529974	Krňany	461	245	216	43,6	42,8	44,6
CZ0201	529991	Křečovice	817	407	410	41,9	40,6	43,3
CZ0201	530000	Křivsoudov	439	231	208	43,0	42,3	43,8

PŘÍLOHA P VI: ŠIKMOST A ŠPIČATOST A JEJÍ ZOBRAZENÍ V HISTOGRAMU A KRABICOVÉM DIAGRAMU



PŘÍLOHA VII: FUNKCE VYBER_EVIDENT_PARTNER_KRITERIA()

```
def vyber_evident_partner_kriteria():
    global vysledek_excel
    global vyber_dat_vysledek
    vysledek_evident =
    hn.vyber_subjektu(hn.Zdrojovy_Kody_Mnozstvi,'Evident_Kraj_Naze
    v',volby_evident_kraj,'Evident_ORP_Nazev', volby_evident_ORP,
    'Evident_ZUJ_Nazev',volby_evident_nazev,'Evident_TypSubjektu',
    volby_evident_typ)

    vysledek_evidentApartner =
    hn.vyber_subjektu(vysledek_evident,'Partner_Kraj_Nazev',volby_
    partner_kraj,'Partner_ORP_Nazev',volby_partner_ORP, 'Part-
    ner_ZUJ_Nazev',volby_partner_nazev, 'Partner_ TypSubjek-
    tu',volby_partner_typ)

    vysledek =
    hn.vyber_kriterii(vysledek_evidentApartner,'Indikator',volby_i
    ndikator,'Kod',volby_kod,'Druh_Odpadu',volby_druhOdpadu,
    'Rok',volby_rok)

    vysledek_excel = vysledek
    vyber_dat_vysledek = vysledek
    if not volby_sloupce:
        vysledek = vysledek.loc[:,volby_sloupce_univ]
    else: vysledek = vysledek.loc[:,volby_sloupce]
    if not vysledek.empty:
        if ‘Odpad_vKg’ in vysledek.columns:
            vysledek[‘Odpad_vKg’] = vysledek[‘Odpad_vKg’]
            .apply(lambda x: locale.format_string("%d", x, grou-
            ping=True))
        if 'Pocet_obyvatel' in vysledek.columns:
            vysledek['Pocet_obyvatel'] = výsledek['Pocet_ obyva-
            tel'].apply(lambda x: locale.format_string("%d", x,
            grouping=True))
        if 'OdpadNaPocetObyv' in vysledek.columns:
            vysledek['OdpadNaPocetObyv'] = výsle-
            dek['OdpadNaPocetObyv'].apply(lambda x: locale.format_
            string("%.2f", x, grouping=True))

        pocet_polozek = len(vysledek.index)
```

```
        text_widget.delete("1.0","end")
        text_widget.insert("1.0",f"VÝPIS FILTROVANÝCH DAT DLE VÝ-
        BĚRU EVIDENTA, PARTNERA A KRITÉRIÍ ({pocet_polozek} polo-
        žek):\n\n {vysledek.to_string(index=False, justi-
        fy='right')}\n")
    else:
        text_widget.delete("1.0","end")
        text_widget.insert("1.0", "Výběr nesplnil žádný zá-
        znam.\n")
```

PŘÍLOHA VIII: FUNKCE SHOW_MAP()

```
def show_map():
    global vyber_dat_vysledek
    global vysledek_excel
    if vyber_dat_vysledek is not None:
        # Načtení geografického souboru
        gdf_kraje = gpd.read_file('kraje-simple.json',
         encoding='utf-8')
        gdf_orp = gpd.read_file('orp-simple.json',
         encoding='utf-8')
        gdf_obce = gpd.read_file('obce-simple.json',
         encoding='utf-8')

        if subjekt_radiobut_value.get() == '1':
            subjekt = 'Evident'
        elif subjekt_radiobut_value.get() == '2':
            subjekt = 'Partner'

        if uzemi_radiobut_value.get() == '1':
            uzemi_cislo = 'Kraj_Cislo'
            mapa = gdf_kraje
            json_column = 'NUTS3_KOD'
            uzemi_nazev = 'Kraj_Nazev'
            nazev_souboru_unique = hn.unique_kraj
        elif uzemi_radiobut_value.get() == '2':
            uzemi_cislo = 'ORP_Cislo'
            mapa = gdf_orp
            json_column = 'ORP_Cislo'
            uzemi_nazev = 'ORP_Nazev'
            nazev_souboru_unique = hn.unique_orp
        elif uzemi_radiobut_value.get() == '3':
            uzemi_cislo = 'ZUJ_Cislo'
            json_column = 'KOD'
            mapa = gdf_obce
            uzemi_nazev = 'ZUJ_Nazev'
            nazev_souboru_unique = hn.LexikonObci

        if sloupecHodnoty_radiobut_value.get() == '1':
            hodnoty = 'OdpadNaObyv_g'
            jednotka = 'g'
        elif sloupecHodnoty_radiobut_value.get() == '2':
```

```
            hodnoty = 'Odpad_vKg'
            jednotka = 'kg'

        nazev_sloupce = f'{subjekt}_{uzemi_cislo}'
        nazev_sloupce_lexikon = uzemi_cislo
        nazev_sloupce_unique_nazev = uzemi_nazev

        mapa_data = hn.odpadNaObyvatele_g(vyber_dat_vysledek,
         nazev_sloupce,hn.LexikonObci,nazev_sloupce_lexikon)

        bezNul_data = mapa_data[mapa_data['Odpad_vKg'] > 0]

        bezNul_data = pd.merge(bezNul_data,
         nazev_souboru_unique[[nazev_sloupce_lexikon,
         nazev_sloupce_unique_nazev]], on=nazev_sloupce_lexikon,
         how='left')

        # zaokrouhlení sloupců na celá čísla nahoru
        bezNul_data['OdpadNaObyv_g'] = bezNul_data
         ['OdpadNaObyv_g'].apply(np.ceil).astype(int)
        bezNul_data['Odpad_vKg'] = bezNul_data
         ['Odpad_vKg'].apply(np.ceil).astype(int)

        #sloučení df kraje_produkce s geometrickým df podle
         názvu kraje
        gdf_merged = mapa.merge(mapa_data, left_on=
         json_column, right_on=nazev_sloupce,how='left')

        # nahrazení chybějících hodnot v datovém rámci gdf_merged
        gdf_merged['OdpadNaObyv_g'] = gdf_merged
         ['OdpadNaObyv_g'].fillna(value=0)
        gdf_merged['Odpad_vKg'] = gdf_merged
         ['Odpad_vKg'].fillna(value=0)

        # zaokrouhlení sloupce hodnoty ('OdpadNaObyvatele')
         na celá čísla nahoru
        gdf_merged['OdpadNaObyv_g'] =
gdf_merged['OdpadNaObyv_g'].apply(np.ceil).astype(int)
        gdf_merged['Odpad_vKg'] =
gdf_merged['Odpad_vKg'].apply(np.ceil).astype(int)
```

```
        bezNul_data = bezNul_data.sort_values(by=hodnoty, ascen-
ding=True)
        gdf_merged_filtered =
gdf_merged.loc[gdf_merged['Odpad_vKg'] > 0].sort_values(hodnoty)

        spodni_hranice, horni_hranice =
hn.zjisteni_hranic(gdf_merged_filtered,hodnoty)

        outliers =
gdf_merged_filtered[(gdf_merged_filtered[hodnoty] < spod-
ni_hranice) | (gdf_merged_filtered[hodnoty] > horni_hranice)]

        pocet_odlehlych_hodnot = len(outliers)
        if pocet_odlehlych_hodnot > 0:
            upper_limit_scale = horni_hranice
        else:
            upper_limit_scale = gdf_merged_filtered[hodnoty].max()

        format_column(bezNul_data)
        text_widget.delete("1.0","end")
        text_widget.insert(END, "Data pro vznik mapy:\n\n ")

        mapa_sloupce = [nazev_sloupce_unique_nazev, na-
zev_sloupce_lexikon,'OdpadNaObyv_g','Pocet_Obyvatel','Odpad_vKg']
        text_widget.insert(END, bez-
Nul_data[mapa_sloupce].to_string(index=False,justify='left'))

        vysledek_excel =bezNul_data[mapa_sloupce]

        cmap = cm.get_cmap('viridis')
        cmap = cmap.reversed()
        cmap.set_over('black')
        cmap.set_under('white')

        # nastavení rozsahu hodnot pro barevnou mapu
        vmin = 1
        vmax = upper_limit_scale  # nastavíme max hodnotu v hodno-
tách jako vrchol legendy
        norm = plt.Normalize(vmin=vmin, vmax=vmax)

        fig, ax = plt.subplots()
```

```
        # použití metody plot() pro zobrazení mapy
        gdf_merged.plot(column = hodnoty,
                        cmap = cmap,
                        ax=ax,
                        norm=norm,
                        legend = True,
                        edgecolor='black',
                        linewidth=0.5,
                        alpha=0.8,
                        )

        if len(gdf_merged[gdf_merged[hodnoty] == 0]) > 0:
            text_bila_mista = 'Bílá místa znázorňují území, která
neevidovala tento druh odpadu. '
        else: text_bila_mista = ''
        if pocet_odlehlych_hodnot > 0:
            text_odlehle_hodnoty = 'Černě jsou zvýrazněny\n vybo-
čující hodnoty nad {} '
            uvedeni_jednotky = f'{jednotka}'
            tecka = '.'
        else:
            text_odlehle_hodnoty = ''
            uvedeni_jednotky = ''
            tecka = ''

        # popisky názvů míst nezobrazovat na úrovni ZÚJ
        if uzemi_radiobut_value.get() != '3':
            for index, row in gdf_merged.iterrows():
              plt.annotate(text=row['NAZEV'],
                      xy=row['geometry'].centroid.coords[0],
                      horizontalalignment='center',
                      color='black',
                      fontsize=5)

        horni_hranice_zaokr = zaokrouhleni(horni_hranice)


        # Přidání textu s hodnotou vmax
        formatted_upper_bound =
'{:,.0f}'.format(horni_hranice_zaokr).replace(',', ' ')

ax.annotate(f'{text_odlehle_hodnoty}{uvedeni_jednotky}{tecka}\n{te
```

```
xt_bila_mista}'.format(formatted_upper_bound), xy=(0.95, 0.1), xy-
coords='axes fraction', ha='right', va='center')
        # Přidání jednotky nad barevnou škálu
        ax.text(1.1, 1.15, f'{jednotka}', transform=ax.transAxes,
        fontsize=12, color='black', ha='center')

        if sloupecHodnoty_radiobut_value.get() == '1':
            hodnoty_text = 'Mapa zobrazuje hodnoty:   v g na oby-
vatele\n'
        elif sloupecHodnoty_radiobut_value.get() == '2':
            hodnoty_text = 'Mapa zobrazuje hodnoty:   odpad v
kg\n'
        if subjekt_radiobut_value.get() == '1':
            subjekt_text = 'Nakládání s odpady z pohledu:   evi-
dentů \n'
        elif subjekt_radiobut_value.get() == '2':
            subjekt_text = 'Nakládání s odpady z pohledu:   part-
nerů \n'
        if uzemi_radiobut_value.get() == '1':
            uzemi_text = 'Shrnutí na úrovni:   krajů\n'
        elif uzemi_radiobut_value.get() == '2':
            uzemi_text = 'Shrnutí na úrovni:   ORP\n'
        elif uzemi_radiobut_value.get() == '3':
            uzemi_text = 'Shrnutí na úrovni:   ZÚJ\n'
        if volby_evident_kraj:
            text = create_title_from_list(volby_evident_kraj)
            evident_kraj = f'Evident kraj:   {text}      '
        else: evident_kraj= ''
        if volby_evident_ORP:
            text = create_title_from_list(volby_evident_ORP)
            evident_ORP = f'Evident ORP:   {text}      '
        else: evident_ORP= ''
        if volby_evident_nazev:
            text = create_title_from_list(volby_evident_nazev)
            evident_ZUJ = f'Evident ZÚJ:   {text}      '
        else: evident_ORP= ''
        if volby_evident_typ:
            text = create_title_from_list(volby_evident_typ)
            evident_typ = f'Evident typ subjektu:   {text}'
        else: evident_typ= ''
        if volby_partner_kraj:
            text = create_title_from_list(volby_partner_kraj)
```

```
            partner_kraj = f'\nPartner kraj:   {text}      '
        else: partner_kraj= ''
        if volby_partner_ORP:
            text = create_title_from_list(volby_partner_ORP)
            partner_ORP = f' Partner ORP:   {text}      '
        else: partner_ORP= ''
        if volby_partner_nazev:
            text = create_title_from_list(volby_partner_nazev)
            partner_ZUJ = f' Partner ZÚJ:   {text}      '
        else: partner_ZUJ= ''
        if volby_partner_typ:
            text = create_title_from_list(volby_partner_typ)
            partner_typ = f' Partner typ subjektu:   {text}'
        else: partner_typ= ''
        if volby_druhOdpadu:
            text = create_title_from_list(volby_druhOdpadu)
            odpad = f'\nDruh odpadu:   {text}'
        else: odpad= ''
        if volby_indikator:
            text = create_title_from_list(volby_indikator)
            indikator = f'\nIndikátor:   {text}'
        else: indikator = ''
        if volby_kod:
            text = create_title_from_list(volby_kod)
            nakladani = f'\nKód způsobu nakládání:   {text}'
        else: nakladani = ''
        if volby_rok:
          text = create_title_from_list(volby_rok)
          obdobi = f'\nObdobí:   {text}'
        else: obdobi = ''
        # Sestavení titulku mapy podle voleb uživatele
        title_text =
f'{hodnoty_text}{subjekt_text}{uzemi_text}{evident_kraj}{evident_O
RP}{evident_ZUJ}{evident_typ}{partner_kraj}{partner_ORP}{partner_Z
UJ}{partner_typ}{odpad}{indikator}{nakladani}{obdobi}'
        plt.title(title_text,ha='left',loc='left',fontsize=10)

        plt.show()
    else:
        messagebox.showwarning("Chyba", "Nebyla nalezena žádná da-
ta k zobrazení v mapě.")
```

PŘÍLOHA IX: FUNKCE RELATIVNI_CETNOSTI()

```
def relativni_cetnosti():
    global vysledek_excel
    global vyber_dat_vysledek
    if vyber_dat_vysledek is not None:
        if (volby_evident_kraj) and (not (volby_evident_ORP or
volby_evident_nazev)):
            nazev_souboru_unique = hn.unique_orp
            nazev_sloupce_lexikon = 'ORP_Cislo'
            nazev_sloupce_unique_nazev = 'ORP_Nazev'
            column_grouped = 'Evident_ORP_Cislo'
            popisek_osaX = 'ORP'
        elif (volby_evident_ORP) and (not (volby_evident_kraj or
volby_evident_nazev)):
            nazev_souboru_unique = hn.LexikonObci
            nazev_sloupce_lexikon = 'ZUJ_Cislo'
            nazev_sloupce_unique_nazev = 'ZUJ_Nazev'
            column_grouped = 'Evident_ZUJ_Cislo'
            popisek_osaX = 'ZÚJ'
        else :
            nazev_souboru_unique = hn.unique_kraj
            nazev_sloupce_lexikon = 'Kraj_Cislo'
            nazev_sloupce_unique_nazev = 'Kraj_Nazev'
            column_grouped = 'Evident_Kraj_Cislo'
            popisek_osaX = 'Kraje'

        vysledek =
hn.odpadNaObyvatele_g2(vyber_dat_vysledek,column_grouped,hn.Lexiko
nObci,nazev_sloupce_lexikon)

        vysledek = pd.merge(vysledek, na-
zev_souboru_unique[[nazev_sloupce_lexikon, na-
zev_sloupce_unique_nazev]], on=nazev_sloupce_lexikon, how='left')

        cetnosti_sloupce = [nazev_sloupce_unique_nazev, na-
zev_sloupce_lexikon,'OdpadNaObyv_g','Pocet_Obyvatel','Odpad_vKg',
'Relativni_cetnost']

        soucet = vysledek['OdpadNaObyv_g'].sum()
        vysledek['Relativni_cetnost'] = vysledek['OdpadNaObyv_g']
/ soucet
```

```
        vysledek = vysledek.sort_values('OdpadNaObyv_g', ascen-
ding=True)

        vysledek_graf = vysledek[cetnosti_sloupce]

        vysledek['Relativni_cetnost'] = vysle-
dek['Relativni_cetnost'].apply('{:.2%}'.format)

        vysledek_excel =vysledek[cetnosti_sloupce]

        format_column(vysledek)

        text_widget.delete("1.0","end")
        text_widget.insert(END, "Procentuelní zastoupení jednotli-
vých krajů:\n\n ")
        text_widget.insert(END, vysle-
dek[cetnosti_sloupce].to_string(index=False,justify='left'))

        fig, ax = plt.subplots(figsize=(12,6))

        ax.bar(vysledek_graf[nazev_sloupce_unique_nazev], vysle-
dek_graf['Relativni_cetnost'], color=cm.viridis(np.linspace(0, 1,
len(vysledek_graf))))

ax.set_xticklabels(vysledek_graf[nazev_sloupce_unique_nazev], ro-
tation=90)
        ax.set_xlabel(popisek_osaX,fontsize=13)
        ax.set_ylabel('Relativní četnosti',fontsize=13)
        # přidání hodnot
        for i, v in enumerate(vysledek_graf['Relativni_cetnost']):
            plt.text(i, v, round(v, 2), color='black',
ha='center', fontsize=10)

        # Sestavení titulku grafu podle voleb uživatele
        title_text = tvorba_popisku_grafu('cast')
        plt.title(title_text,ha='left',loc='left',fontsize=10)

        ax.text(0.95, 1.15, 'Relativní četnosti', trans-
form=ax.transAxes, fontsize=14,
        verticalalignment='top', horizontalalignment='right')
```

```
        ax.text(0.95, 1.10, '\n(sestaveno z hodnot ´odpad na oby-
vatele´)', transform=ax.transAxes, fontsize=10,
        verticalalignment='top', horizontalalignment='right')

        plt.tight_layout()
        plt.show()

    else:
        messagebox.showwarning("Chyba", "Nebyla nalezena žádná da-
ta k zobrazení.")
```

PŘÍLOHA X: FUNKCE HISTOGRAM()

```
def histogram():
    global vysledek_excel
    global vyber_dat_vysledek
    if vyber_dat_vysledek is not None:
        nazev_souboru_unique = hn.LexikonObci
        nazev_sloupce_lexikon = 'ZUJ_Cislo'
        nazev_sloupce_unique_nazev = 'ZUJ_Nazev'
        column_grouped = 'Evident_ZUJ_Cislo'

        vysledek =
hn.odpadNaObyvatele_g2(vyber_dat_vysledek,column_grouped,hn.Lexiko
nObci,nazev_sloupce_lexikon)

        vysledek = pd.merge(vysledek, na-
zev_souboru_unique[[nazev_sloupce_lexikon, na-
zev_sloupce_unique_nazev]], on=nazev_sloupce_lexikon, how='left')

        widget_sloupce = [nazev_sloupce_unique_nazev, na-
zev_sloupce_lexikon,'OdpadNaObyv_g','Pocet_Obyvatel','Odpad_vKg']

        vysledek = vysledek.sort_values('OdpadNaObyv_g', ascen-
ding=True)

        spodni_hranice, horni_hranice =
hn.zjisteni_hranic(vysledek,'OdpadNaObyv_g')

        vysledek_lower = vysledek[vysledek['OdpadNaObyv_g'] <=
horni_hranice]
        vysledek_higher = vysledek[vysledek['OdpadNaObyv_g'] >
horni_hranice]
        pocty_higher = len(vysledek_higher)
        pocty_lower = len(vysledek_lower)
        minimum = vysledek_lower['OdpadNaObyv_g'].min()
        maximum = vysledek_lower['OdpadNaObyv_g'].max()
        maximumVyboc = vysledek_higher['OdpadNaObyv_g'].max()
        prumer = vysledek_lower['OdpadNaObyv_g'].mean()
        median = vysledek_lower['OdpadNaObyv_g'].median()

        vysledek_graf = vysledek_lower
        vysledek_excel =vysledek[widget_sloupce]
```

```
        # Zjištění počtu intervalů podle Sturgesova pravidla
        n = pocty_lower
        k = round(1 + 3.322 * math.log10(n))

        fig, ax = plt.subplots(figsize=(12,6))
        plt.hist(vysledek_graf['OdpadNaObyv_g'],bins=k)
        plt.xlabel('Odpad na obyvatele v gramech')
        plt.ylabel('Počet ZÚJ')

        text_widget.delete("1.0","end")
        text_widget.insert(END, "Histogram četností:\n\n ")
        if pocty_higher > 0:
            format_column(vysledek_higher)
            text_widget.insert(END, f"Záznamy ({pocty_higher}),
které obsahují vybočující hodnoty ve sloupci 'OdpadNaObyv_g':\n\n
")
            text_widget.insert(END, vysle-
dek_higher[widget_sloupce].to_string(index=False,justify='left'))
            text_widget.insert(END, f"\n\nZáznamy ({pocty_lower})
bez vybočujících hodnot, pro zobrazení v histogramu: \n\n ")
        format_column(vysledek_lower)
        text_widget.insert(END, vysle-
dek_lower[widget_sloupce].to_string(index=False,justify='left'))

        # Sestavení popisku min a max hodnoty
        minimum_format = locale.format_string("%d",
round(minimum), grouping=True)
        maximum_format = locale.format_string("%d",
round(maximum), grouping=True)
        ax.annotate(f'minimum: {minimum_format} g, maximum: {ma-
ximum_format} g, počet hodnot {pocty_lower}', xy=(0.95, 0.95), xy-
coords='axes fraction', ha='right', va='center')

        # Sestavení popisku průměr a medián
        prumer_format = locale.format_string("%d", round(prumer),
grouping=True)
        median_format = locale.format_string("%d", round(median),
grouping=True)
        ax.annotate(f'průměr: {prumer_format} g, medián: {medi-
an_format} g', xy=(0.95, 0.90), xycoords='axes fraction',
ha='right', va='center')
```

```
        # Sestavení popisku grafu
        ax.text(0.95, 1.15, 'Histogram četností', trans-
form=ax.transAxes, fontsize=14,
        verticalalignment='top', horizontalalignment='right')

        if pocty_higher > 0:
            horni_hranice_zaokr = zaokrouhleni(horni_hranice)
            horni_hranice_format = locale.format_string("%d",
round(horni_hranice_zaokr), grouping=True)
            ax.text(0.95, 1.1, f'\n(sestaven bez vybočujících hod-
not větších než {horni_hranice_format} gramů )', trans-
form=ax.transAxes, fontsize=10,verticalalignment='top', horizonta-
lalignment='right')
            maximumVyboc_format = locale.format_string("%d",
round(maximumVyboc), grouping=True)
            ax.annotate(f'maximum z vybočujících hodnot {maximu-
mVyboc_format} g, počet vybočujících hodnot {pocty_higher}',
xy=(0.95, 0.85), xycoords='axes fraction', ha='right',
va='center')

        # Sestavení titulku grafu podle voleb uživatele
        title_text = tvorba_popisku_grafu('cast')
        plt.title(title_text,ha='left',loc='left',fontsize=10)
        plt.tight_layout()
        plt.show()
    else:
        messagebox.showwarning("Chyba", "Nebyla nalezena žádná da-
ta k zobrazení.")
```

PŘÍLOHA XI: FUNKCE BOXPLOT()

```
def boxplot():
    global vysledek_excel
    global vyber_dat_vysledek
    if vyber_dat_vysledek is not None:
        nazev_souboru_unique = hn.LexikonObci
        nazev_sloupce_lexikon = 'ZUJ_Cislo'
        nazev_sloupce_unique_nazev = 'ZUJ_Nazev'
        column_grouped = 'Evident_ZUJ_Cislo'

        vysledek =
hn.odpadNaObyvatele_g2(vyber_dat_vysledek,column_grouped,hn.Lexiko
nObci,nazev_sloupce_lexikon)

        vysledek = pd.merge(vysledek, na-
zev_souboru_unique[[nazev_sloupce_lexikon, na-
zev_sloupce_unique_nazev,'Kraj_Nazev','ORP_Nazev']],
on=nazev_sloupce_lexikon, how='left')

        vysledek = vysledek.sort_values('OdpadNaObyv_g', ascen-
ding=True)

        spodni_hranice, horni_hranice =
hn.zjisteni_hranic(vysledek,'OdpadNaObyv_g')

        vysledek_lower = vysledek[vysledek['OdpadNaObyv_g'] <=
horni_hranice]
        vysledek_higher = vysledek[vysledek['OdpadNaObyv_g'] >
horni_hranice]
        pocty_vysledek = len(vysledek)
        pocty_higher = len(vysledek_higher)
        pocty_lower = len(vysledek_lower)

        box_sloupce =
['Kraj_Nazev','ORP_Nazev',nazev_sloupce_unique_nazev, na-
zev_sloupce_lexikon,'OdpadNaObyv_g','Pocet_Obyvatel','Odpad_vKg']

        vysledek_graf = vysledek_lower[box_sloupce]
        vysledek_excel =vysledek_lower[box_sloupce]
```

```
        vysledek_widget = vysle-
dek_lower.sort_values(by=['Kraj_Nazev','ORP_Nazev','OdpadNaObyv_g'
], ascending=[True, True, True])
        format_column(vysledek_widget)

        text_widget.delete("1.0","end")
        text_widget.insert(END, f"Data pro sestavení krabicových
diagramů:\n\n ")
        if pocty_higher > 0:
            vysledek_higher = vysle-
dek_higher.sort_values(by=['Kraj_Nazev','ORP_Nazev','OdpadNaObyv_g
'], ascending=[True, True, True])
            format_column(vysledek_higher)
            text_widget.insert(END, f"Záznamy ({pocty_higher}),
které obsahují vybočující hodnoty ve sloupci 'OdpadNaObyv_g':\n\n
")
            text_widget.insert(END, vysle-
dek_higher[box_sloupce].to_string(index=False,justify='left'))
            text_widget.insert(END, f"\n\nZáznamy ({pocty_lower})
bez vybočujících hodnot, pro zobrazení v krabicových diagramech:
\n\n ")
        format_column(vysledek_lower)
        text_widget.insert(END, vysle-
dek_widget[box_sloupce].to_string(index=False,justify='left'))

        # Seznam názvů všech krajů v datasetu
        kraje = vysledek_graf['Kraj_Nazev'].unique()

        # Seřazení názvů krajů abecedně
        kraje = sorted(kraje)

        # Vytvoření seznamu datových souborů odpadu pro jednotlivé
kraje
        odpady_kraje = []
        for kraj in kraje:
            odpad_kraje = vysle-
dek_graf.loc[vysledek_graf['Kraj_Nazev'] == kraj, 'OdpadNaObyv_g']
            odpady_kraje.append(odpad_kraje)

        # Vytvoření boxplotu pro jednotlivé kraje
        fig, ax = plt.subplots()
        ax.boxplot(odpady_kraje)
```

```
        # Sestavení popisku grafu
        ax.text(0.95, 1.15, 'Krabicové diagramy', trans-
form=ax.transAxes, fontsize=14,
        verticalalignment='top', horizontalalignment='right')

        if pocty_higher > 0:
            horni_hranice_zaokr = zaokrouhleni(horni_hranice)
            horni_hranice_format = locale.format_string("%d",
round(horni_hranice_zaokr), grouping=True)
            ax.text(0.95, 1.1, f'\ngraf sestaven bez vybočujících
hodnot větších než {horni_hranice_format} g', trans-
form=ax.transAxes, fontsize=10,verticalalignment='top', horizonta-
lalignment='right')

        # Sestavení titulku grafu podle voleb uživatele
        title_text = tvorba_popisku_grafu('cast')
        plt.title(title_text,ha='left',loc='left',fontsize=10)

        # Nastavení popisků os a názvu grafu
        ax.set_xlabel('Kraje')
        ax.set_ylabel('Odpad na obyvatele v g')
        # Nastavení popisků na ose x
        ax.set_xticklabels(kraje, rotation=45, ha='right')
        # Nastavení formátu osy y
        formatter = ticker.ScalarFormatter(useOffset=False)
        ax.yaxis.set_major_formatter(formatter)

        # Zobrazení grafu
        plt.tight_layout()
        plt.show()

    else:
        messagebox.showwarning("Chyba", "Nebyla nalezena žádná da-
ta k zobrazení.")
```

PŘÍLOHA XII: FUNKCE ODLEHLE_HODNOTY()

```
def odlehle_hodnoty():
    global vysledek_excel
    global vyber_dat_vysledek
    if vyber_dat_vysledek is not None:
        nazev_souboru_unique = hn.LexikonObci
        nazev_sloupce_lexikon = 'ZUJ_Cislo'
        nazev_sloupce_unique_nazev = 'ZUJ_Nazev'
        column_grouped = 'Evident_ZUJ_Cislo'

        vysledek =
hn.odpadNaObyvatele_g2(vyber_dat_vysledek,column_grouped,hn.Lexiko
nObci,nazev_sloupce_lexikon)

        vysledek = pd.merge(vysledek, na-
zev_souboru_unique[[nazev_sloupce_lexikon, na-
zev_sloupce_unique_nazev,'Kraj_Nazev','ORP_Nazev']],
on=nazev_sloupce_lexikon, how='left')

        vysledek = vysledek.sort_values('OdpadNaObyv_g', ascen-
ding=True)

        pocet_hodnot = len(vysledek['OdpadNaObyv_g'])
        perc_lower=0.25
        perc_higher1 = 0.95
        perc_higher = 0.75
        lower_bound1, upper_bound1 =
hn.zjisteni_hranic2(vysledek,'OdpadNaObyv_g',perc_lower,perc_highe
r1)
        lower_bound, upper_bound =
hn.zjisteni_hranic2(vysledek,'OdpadNaObyv_g',perc_lower,perc_highe
r)
        outliers1 = vysledek[(vysledek['OdpadNaObyv_g'] <
lower_bound1) | (vysledek['OdpadNaObyv_g'] > upper_bound1)]
        outliers = vysledek[(vysledek['OdpadNaObyv_g'] <
lower_bound) | (vysledek['OdpadNaObyv_g'] > upper_bound)]
        po-
cet_odlehlych_hodnot_lower=len(vysledek[(vysledek['OdpadNaObyv_g']
< lower_bound)])
```

```
        po-
cet_odlehlych_hodnot_upper=len(vysledek[(vysledek['OdpadNaObyv_g']
> upper_bound)])
        vysledek_lowers1 = vysledek[(vysledek['OdpadNaObyv_g'] <
upper_bound1)]
        pocet_odlehlych_hodnot = len(outliers)

        vysledek_excel = outliers

        widget_sloupce = [nazev_sloupce_unique_nazev, na-
zev_sloupce_lexikon,'OdpadNaObyv_g','Pocet_Obyvatel','Odpad_vKg']
        format_column(outliers)
        format_column(outliers1)
        # Výsledky testu
        if len(outliers) > 0:
            vysledek_testu_text = f"V datovém vzorku o {po-
cet_hodnot} hodnotách existuje celkem {pocet_odlehlych_hodnot} od-
lehlých hodnot, které přesáhly horní hranici {perc_higher} + IQR x
1,5:\n\n{outliers[widget_sloupce].to_string(index=False, justi-
fy='right')}"
            print(outliers)
            if len(outliers1) > 0:
                vysledek_testu_text1 = f"V datovém vzorku se vy-
skytly  hodnoty ({len(outliers1)}), které přesáhly horní hranici
{perc_higher1} + IQR x 1,5  a v grafu nejsou zobrazeny.\nJedná se
o tyto hodno-
ty:\n\n{outliers1[widget_sloupce].to_string(index=False, justi-
fy='right')}\n\n"
            else:
                vysledek_testu_text1=""
        else:
            vysledek_testu_text = f"Odlehlé hodnoty v datovém
vzorku o {pocet_hodnot} hodnotách nejsou zjištěny."
            vysledek_testu_text1=""
        text_widget.delete("1.0","end")
        text_widget.insert(END, "DIXONŮV TEST PRO ODHALENÍ ODLEH-
LÝCH HODNOT:\n\n ")
        text_widget.insert(END, vysledek_testu_text1)
        text_widget.insert(END, vysledek_testu_text)

        # Tvorba grafu Scatter plot
```

```
        perc_25 =
np.percentile(vysledek_lowers1['OdpadNaObyv_g'],25)
        perc_50 =
np.percentile(vysledek_lowers1['OdpadNaObyv_g'],50)
        perc_75 =
np.percentile(vysledek_lowers1['OdpadNaObyv_g'],75)

        data = vysledek_lowers1['OdpadNaObyv_g']
        fig, ax = plt.subplots()
        ax.scatter(range(len(data)), data)
        ax.axhline(0, color='black', linestyle='solid')
        ax.axhline(upper_bound, color='orange', linesty-
le='solid',label = 'q3 + 1,5 x IQR')

        ax.axhline(perc_25, color='grey', linestyle='--', la-
bel='perc. 25')
        ax.axhline(perc_50, color='green', linestyle='--', label =
'perc. 50')
        ax.axhline(perc_75, color='blue', linestyle='--', label =
'perc. 75')

        ax.set_xlabel('Index',fontsize=14)
        ax.set_ylabel('Odpad na obyvatele (g)', fontsize=14)
        ax.legend()
        # Sestavení titulku grafu podle voleb uživatele
        title_text = tvorba_popisku_grafu('cast')
        plt.title(title_text,ha='left',loc='left',fontsize=12)

        # Sestavení popisku grafu
        ax.text(0.95, 1.2, 'Diagram rozptýlení', trans-
form=ax.transAxes, fontsize=16,
        verticalalignment='top', horizontalalignment='right')

        if len(outliers1) > 0:
            horni_hranice_zaokr = zaokrouhleni(upper_bound1)
            horni_hranice_format = locale.format_string("%d",
round(horni_hranice_zaokr), grouping=True)
            ax.text(0.95, 1.15, f'\ngraf sestaven bez vybočujících
hodnot ({len(outliers1)})  větších než {horni_hranice_format} g',
transform=ax.transAxes, fontsize=12,verticalalignment='top', hori-
zontalalignment='right')
```

```
            plt.tight_layout()
            plt.show()

        else:
            messagebox.showwarning("Chyba", "Nebyla nalezena žádná da-
ta k zobrazení.")
```

PŘÍLOHA XIII: FUNKCE SUMARIZACE()

```
def sumarizace():
    global vysledek_excel
    global vyber_dat_vysledek
    if vyber_dat_vysledek is not None:
        if volby_evident_nazev:
            text_widget.insert(END, f"\n\nPozor charakteristiky
lze vypočítat pouze pro úroveň krajů nebo ORP.\n\n ")
        else:
            nazev_souboru_unique = hn.LexikonObci
            nazev_sloupce_lexikon = 'ZUJ_Cislo'
            nazev_sloupce_unique_nazev = 'ZUJ_Nazev'
            column_grouped = 'Evident_ZUJ_Cislo'

            vysledek =
hn.odpadNaObyvatele_g2(vyber_dat_vysledek,column_grouped,hn.Lexiko
nObci,nazev_sloupce_lexikon)

            vysledek = pd.merge(vysledek, na-
zev_souboru_unique[[nazev_sloupce_lexikon, na-
zev_sloupce_unique_nazev,'Kraj_Nazev','ORP_Nazev']],
on=nazev_sloupce_lexikon, how='left')

            box_sloupce =
['Kraj_Nazev','ORP_Nazev',nazev_sloupce_unique_nazev, na-
zev_sloupce_lexikon,'OdpadNaObyv_g','Pocet_Obyvatel','Odpad_vKg']

            # Zde se nastavouje, jaká data půjdou do modelu
            vysledek_graf = vysledek[box_sloupce]

            if volby_evident_kraj:
                nazev_sloupce = 'ORP'
                uzemi_sloupec = 'ORP_Nazev'
                nazev_uzemi = 'ORP'
            elif volby_evident_ORP:
                nazev_sloupce = 'ORP'
                uzemi_sloupec = 'ORP_Nazev'
                nazev_uzemi = 'ORP'
            else:
                nazev_sloupce = 'Kraj'
                uzemi_sloupec = 'Kraj_Nazev'
```

```
            nazev_uzemi = 'krajích'

            # Vytvoření prázdného dataframe pro výsledky výpočtu
charakteristik polohy
            metriky_df =
pd.DataFrame(columns=[nazev_sloupce,'Pocet_hodnot', 'Prumer', 'Me-
dian','Minimum','25.percentil', '50.percentil',
'75.percentil','Maximum', 'Smerodatna_odchylka'])

            # Seznam názvů všech krajů v datasetu
            uzemi_unique = vysledek_graf[uzemi_sloupec].unique()

            # Seřazení názvů krajů abecedně
            uzemi_unique = sorted(uzemi_unique)

            # Vytvoření seznamu datových souborů odpadu pro jed-
notlivé území
            odpady_seznam = []
            for i in uzemi_unique:
                odpad = vysle-
dek_graf.loc[vysledek_graf[uzemi_sloupec] == i, 'OdpadNaObyv_g']
                odpady_seznam.append(odpad)

            # Výpočet charakteristik pro jednotlivé kraje
            charakteristiky = []
            for odpad in odpady_seznam:
                minimum = odpad.min()
                maximum = odpad.max()
                pocet_hodnot = odpad.count()
                prumer = odpad.mean()
                median = odpad.median()
                perc_25 = np.percentile(odpad, 25)
                perc_50 = np.percentile(odpad, 50)
                perc_75 = np.percentile(odpad, 75)
                sm_odchylka = odpad.std(ddof=1)
                charakteristiky.append([pocet_hodnot, prumer, me-
dian, minimum, perc_25, perc_50, perc_75, maximum, sm_odchylka])

            # Vytvoření dataframe s výpočty
            metriky_df = pd.DataFrame(charakteristiky, co-
lumns=['Pocet_hodnot', 'Prumer', 'Medi-
```

```
an','Minimum','25.percentil', '50.percentil',
'75.percentil','Maximum', 'Smerodatna_odchylka'])

            # Vložení sloupce s názvy krajů do prvního sloupce df
            metriky_df.insert(0, nazev_sloupce, uzemi_unique)
            vysledek_excel = metriky_df
            # TVORBA TEXT WIDGETU
            # Charakteristiky polohy
            format_column(metriky_df)
            text_widget.delete("1.0","end")
            text_widget.insert(END, f"\nCharakteristiky polohy od-
padu na obyvatele v jednotlivých {nazev_uzemi} (g): \n\n")
            text_widget.insert(END,f"Indikátor: {volby_indikator}
  Kód nakládání: {volby_kod} \nDruh odpadu: {volby_druhOdpadu}
Rok: {volby_rok}\nKraj: {volby_evident_kraj}\nORP: {vol-
by_evident_ORP}  \n\n ")

            text_widget.insert('end', metri-
ky_df.to_string(index=False))

    else:
        messagebox.showwarning("Chyba", "Nebyla nalezena žádná da-
ta k zobrazení.")
```