Lekce 13: Geoprocessingové nástroje II

Python pro GIS - Rastrové analýzy

Vojtěch Barták, FŽP ČZU Praha

2025-11-24

Table of contents

1 Data a nastavení

Data ke stažení:

- `dtm_Liberec/` - složka s rastrem digitálního modelu terénu
- `obce/` - složka s polygony obcí v ČR
- `dtm_katastry/` - složka s DTM pro jednotlivé katastry
- `CLC_CR_2018/` - Složka s rastrem Corine Land Cover pro ČR
- `data_vinice/` - složka s vrstevnicemi pro závěrečnou úlohu

Nastavení prostředí:

```
import arcpy

# Workspace
arcpy.env.workspace = r"C:\TEMP\Python_GIS\Lekce_13"
arcpy.env.overwriteOutput = True

# Aktivace Spatial Analyst
arcpy.CheckOutExtension("Spatial")
```

> **i** Pracovní prostředí
>
> **Úlohy 1-11** řešte v **Python Window** v ArcGIS Pro.
> **Úlohy 12-14** (dávkové zpracování) řešte v **Python Notebooku**.

---

2 Analýza terénu

2.1 Úloha 1: Svažitost

Zadání: Spočítejte svažitost (v procentech).

```
# Výpočet svažitosti
slope = arcpy.sa.Slope("dtm_Liberec.tif", "PERCENT_RISE")

# Výsledek existuje pouze v paměti!
# Pro uložení na disk použijte metodu save()
slope.save("svazitost")
```

! Rastrové nástroje a paměť

Rastrové nástroje vrací **Raster objekt**, který existuje pouze v **operační paměti**. Pro trvalé uložení na disk musíte použít metodu `.save()`:

```
# Pouze v paměti:
slope = arcpy.sa.Slope("dtm_Liberec.tif", "PERCENT_RISE")

# Uložení na disk:
slope.save("svazitost")
```

Do metody `.save()` lze zadat buď absolutní cestu, nebo jen relativní (viz ukázka). V druhém případě se rastr uloží do aktuálního pracovního workspace.
Pokud výstupní rastr nepotřebujete uchovat, ale jen použít v dalších nástrojích v rámci stejné analýzy, není třeba rastr ukládat na disk. Lze použít proměnnou s Raster objektem vytvořeným z předchozí analýzy.

Vizuální kontrola: Otevřete vrstvu `svazitost` v Contents a zkontrolujte rozsah hodnot.

2.2 Úloha 2: Aspekt

Zadání: Spočítejte aspekt.

```
# Výpočet aspektu (orientace svahu)
aspect = arcpy.sa.Aspect("dtm_Liberec.tif")
```

Interpretace hodnot:

- 0° / 360° = sever
- 90° = východ
- 180° = jih
- 270° = západ
- -1 = rovné plochy

---

3 Lokální mapová algebra

3.1 Úloha 3: Svažitost $>$ 5 %

Zadání: Najděte místa, kde je svažitost větší než 5 %.

3.1.1 Způsob 1: Nástroj GreaterThan

```
# Použití funkce GreaterThan
slope_gt5 = arcpy.sa.GreaterThan(slope, 5)
```

3.1.2 Způsob 2: Přímé použití operátorů

```
# Kratší zápis pomocí operátoru >
slope_gt5 = slope > 5
```

V obou případech jsme použili proměnnou `slope` vytvořenou v předchozích krocích.

> Dva způsoby zápisu
>
> **Funkce** (`GreaterThan`, `LessThan`, `BooleanAnd` apod.) - explicitní, čitelné
> **Operátory** (`>`, `<`, `==`) - stručné, pythonické
> Oba způsoby dávají stejný výsledek: rastr s hodnotami 1 (True) a 0 (False).

3.2 Úloha 4: Jižní aspekt

Zadání: Najděte místa, kde je aspekt zhruba jižní (mezi 135° a 225°).

```
# Jižní aspekt: 135° - 225°
aspect_south = (aspect >= 135) & (slope <= 225)
aspect_south.save("aspekt_jizni")
```

Alternativní zápis pomocí funkcí:

```
ge135 = arcpy.sa.GreaterThanEqual("aspekt", 135)
le225 = arcpy.sa.LessThanEqual("aspekt", 225)
aspect_south = ge135 & le225
```

> Tip
>
> V pythonu se pro operátor AND standardně používá `and`. Pro mapovou algebru v rámci ArcPy se ale používá `&`.

3.3 Úloha 5: Multikriteriální analýza

Zadání: Najděte místa, kde jsou splněny obě podmínky (svažitost > 5 % **A** jižní aspekt).

```
# Kombinace obou podmínek pomocí & (AND)
suitable = slope_gt5 & aspect_south
suitable.save("vhodna_mista")
```

Nebo vše v jednom příkazu:

```
suitable = (slope > 5) & (aspect >= 135) & (aspect <= 225)
suitable.save("vhodna_mista_v2")
```

> **i** Interpretace
>
> Výsledek má hodnotu 1 tam, kde jsou splněny **obě** podmínky současně - například vhodná místa pro vinice nebo sluneční panely.
> Pokud bychom chtěli najít místa, kde je splněna **alespoň jedna** z podmínek, použili bychom operátor OR. Ten se v Pythonu standardně píše `or`, pro mapovou algebru v rámci ArcPy se ale používá `|`:
>
> ```
> suitable = (slope > 5) | (aspect >= 135) & (aspect <= 225)
> ```

---

4 Fokální mapová algebra

4.1 Úloha 6: Vyhlazení terénu

Zadání: Zhlaďte model terénu klouzavým průměrem 3×3 buňky.

```
# Focal Statistics - průměr v okolí 3x3 buňky
dtm_smooth = arcpy.sa.FocalStatistics("dtm_Liberec.tif")
dtm_smooth.save("dtm_smooth") # není nutné, pokud nechcete rastr uložit
```

> **i** Note
>
> Nástroj `FocalStatistics` má defaultně nastavené okolí 3x3 buňky a počítá průměr. Níže je ukázka, jak by vypadalo volání nástroje s explicitním nastavením těchto parametrů:

```
dtm_smooth = arcpy.sa.FocalStatistics(
    "dtm_Liberec.tif",
    NbrRectangle(3, 3, "CELL"),
    "MEAN"
)
```

Parametry:

- `NbrRectangle(width, height, "CELL")` - obdélníkové okolí
- `"MEAN"` - průměrná hodnota (další možnosti: MAX, MIN, STD, SUM apod.)

Zkuste změnit okno na 5x5 buněk a spočítat vždy maximální místo průměrné hodnoty.

4.2 Úloha 7: Vrstevnice

Zadání: Vytvořte vrstevnice z původního a zhlazeného modelu terénu (po 200 metrech) a porovnejte je.

```
# Vrstevnice z původního DTM
arcpy.sa.Contour(
    "dtm_Liberec.tif",
    "vrstevnice_original",
    200  # interval vrstevnic
)

# Vrstevnice ze zhlazeného DTM
arcpy.sa.Contour(
    "dtm_smooth",
    "vrstevnice_smooth",
    200
)
```

i Motivace pro vyhlazení

Otevřete obě vrstvy vrstevnic v ArcGIS Pro a porovnejte je:

- **Původní vrstevnice** - více detailů, ale "zubatější"
- **Vyhlazené vrstevnice** - hladší průběh, méně šumu

Vyhlazení je užitečné pro **vizualizaci terénu** nebo když potřebujete **odstranit drobné terénní artefakty** (např. chyby měření).

5 Zonální mapová algebra

5.1 Úloha 8: Převýšení v katastrech

Zadání: Zjistěte převýšení terénu na území jednotlivých obcí (max - min).

```
# Zonal Statistics as Table
arcpy.sa.ZonalStatisticsAsTable(
    "obce.shp",  # zóny
    "NAZEV",                # identifikační pole
    "dtm_Liberec.tif",      # hodnoty
    "stat_katastry",        # výstupní tabulka
    statistics_type="RANGE"
)
```

5.2 Úloha 9: Zastoupení krajinného pokryvu

Zadání: Zjistěte zastoupení jednotlivých tříd krajinného pokryvu v jednotlivých okresech.

```
# Tabulate Area
arcpy.sa.TabulateArea(
    "okresy.shp",       # zóny
    "NAZEV",            # identifikační pole zón
    "CLC_2018.tif",     # kategorie (třídy)
    "VALUE",            # pole s kódy kategorií
    "zastoupeni_clc"    # výstupní tabulka
)
```

Výsledek: Tabulka obsahující rozlohu (v m^2) každé třídy CLC v každém okrese.

> Kdy použít co?
>
> **ZonalStatisticsAsTable** - chci statistiky (průměr, max, min) pro každou zónu
> **TabulateArea** - chci vědět, kolik plochy zaujímá každá **kategorie** v každé zóně

5.3 Úloha 10: Relativní zastoupení lesů

Zadání: Zjistěte relativní zastoupení lesů na území jednotlivých obcí.

5.3.1 Krok 1: Vytvoření binární masky lesů

```
# Lesy v CLC mají kód 3
# Vytvoříme rastr 1/0 (les/neles)
lesy = arcpy.sa.Equal('CLC_2018.tif', 3)
```

Alternativně bychom mohli použít oprátor '==='. K tomu je ale nutné mít vstupní rastr v podobě Raster objektu. Ten vytvoříme funkcí `Raster`:

```
# vytvoření Raster objektu z rastru CLC
clc = arcpy.sa.Raster('CLC_2018.tif')

# vytvoření binární masky lesů
lesy = clc == 3
```

5.3.2 Krok 2: Zonální statistiky

```
# Průměr z masky 1/0 = relativní zastoupení
arcpy.sa.ZonalStatisticsAsTable(
    "obce.shp",
    "NAZEV",
    "lesy_mask",
    "zastoupeni_lesu",
    statistics_type="MEAN"
)
```

> **i** Vysvětlení
>
> Průměr z binární masky (0/1) udává **podíl buněk s hodnotou 1**, tedy relativní zastoupení lesů v zóně.
> Příklad: průměr 0.35 = 35 % plochy okresu tvoří lesy.

---

6 Reklasifikace

6.1 Úloha 11: Výšková pásma

Zadání: Reklasifikujte model terénu na výškové pásy po 200 metrech.

Základem reklasifikace je **reklasifikační tabulka**. Pokud chceme konkrétní původní hodnotě přiřadit konkrétní novou hodnotu, má tato tabulka **dva sloupce** - jeden pro původní a druhý pro cílovou hodnotu. Pokud chceme novou hodnotu přiřazovat nějakému **rozsahu hodnot** z původního rastru (jako je tomu v naší úloze), má tabulka **tři sloupce** - první dva pro dolní a horní mez rozsahu, třetí pro cílovou hodnotu.

V Pythonu se reklasifikační tabulka reprezentuje pomocí objektů `RemapValue` (pro konkrétní hodnoty), nebo `RemapRange` (pro rozsahy hodnot). Do nich se hodnoty předávají jako **seznam seznamů**, ve kterém vnitřní seznamy představují jednotlivé řádky tabulky (viz ukázka).

Rozsah nadmořských výšek našeho DTM je 211 až 1434 m n. m.

```
# Vytvoření remap tabulky
remap_list = [[0,200,1],[200,400,2],[400,600,3],[600,800,4],[800,1000,5],[1000,1200,6],[1200
remap = arcpy.sa.RemapRange(remap_list)

# Reklasifikace
vyskova_pasma = arcpy.sa.Reclassify(
    "dtm_Liberec.tif",
    "VALUE",
    remap
)
vyskova_pasma.save("vyskova_pasma_200m")
```

> Automatické generování reklasifikační tabulky
>
> Ti zkušenější si mohou zjistit rozsah rastru programátorsky a podle toho automaticky nastavit ‘RemapRange“:

```python
# Zjištění rozsahu hodnot DTM
result = arcpy.management.GetRasterProperties("dtm_Liberec.tif", "MINIMUM")
min_elev = int(float(result.getOutput(0)))
result = arcpy.management.GetRasterProperties("dtm_Liberec.tif", "MAXIMUM")
max_elev = int(float(result.getOutput(0)))

# Vytvoření remap tabulky
min_round = (min_elev // 200) * 200
max_round = ((max_elev // 200) + 1) * 200

remap_list = []
for i in range(min_round, max_round, 200):
    remap_list.append([i, i+200, i])

remap = arcpy.sa.RemapRange(remap_list)

# Reklasifikace
vyskova_pasma = arcpy.sa.Reclassify(
    "dtm_Liberec.tif",
    "VALUE",
    remap
)
vyskova_pasma.save("vyskova_pasma_200m")
```

---

7 Dávkové zpracování

> **!** Python Notebook
>
> **Úlohy 12-14 řešte v Python Notebooku**, ne v Python Window. Notebook umožňuje lepší strukturu kódu a dokumentaci procesu.

7.1 Úloha 12: Multikriteriální analýza pro všechny katastry

Zadání: Řešte úlohu 5 (vhodná místa: svažitost > 5 % + jižní aspekt) pro jednotlivé katastry Libereckého kraje.

7.1.1 Python Notebook řešení

```
import arcpy
import os

# Nastavení
workspace = r"C:\TEMP\Python_GIS\Lekce_13\dtm_katastry"
arcpy.env.workspace = workspace
arcpy.env.overwriteOutput = True
arcpy.CheckOutExtension("Spatial")

# Získání seznamu všech DEM
dem_list = arcpy.ListRasters("*.tif")
print(f"Nalezeno {len(dem_list)} katastrů")
print(f"{'='*60}")

# Počítadla
uspesnych = 0
chybnych = 0

# Zpracování každého DEM
for dem in dem_list:
    print(f"Zpracovávám: {dem}")

    try:
        # 1. Svažitost
        slope = arcpy.sa.Slope(dem, "PERCENT_RISE")

        # 2. Aspekt
        aspect = arcpy.sa.Aspect(dem)

        # 3. Vhodná místa (svažitost > 5% + jižní aspekt)
        suitable = (slope > 5) & (aspect >= 135) & (aspect <= 225)

        # 4. Uložení výsledku
        output_name = f"{os.path.splitext(dem)[0]}_suitable"
        suitable.save(output_name)

        print(f"  OK - výstup: {output_name}")
        uspesnych += 1

    except Exception as e:
```

```
        print(f"  CHYBA: {str(e)}")
        chybnych += 1
```

7.2 Úloha 13: Analýza vhodnosti pro vinici

Zadání: Jako vášnivý milovník vína toužíte po vlastní vinici. Nejprve je pro ni však třeba vybrat vhodné území, splňující následující kritéria: bude na svazích se sklonem menším než 20%, s jihovýchodní až jihozápadní orientací a zároveň skryta před zraky nenechavých turistů, kteří by ji mohli z okolních vyhlídek vidět a přijít na ochutnávku hroznů. Jakou rozlohu má vhodné území pro založení vinice? Řešte pro jeden katastr.

Kritéria:

1. Svah se sklonem < 20 %
2. Jihovýchodní až jihozápadní orientace (135-225°)
3. Skryto před vyhlídkami (analýza viditelnosti)

Vstup: vrstevnice (SHP)

7.2.1 Krok 1: Vytvoření DTM z vrstevnic

```
import arcpy

# Nastavení
arcpy.env.workspace = r"C:\TEMP\Python_GIS\Lekce_13\data_vinice"
arcpy.CheckOutExtension("Spatial")

# Vstup vrstevnic do TopoToRaster
# formát: [vrstva, pole s výškou]
input_contours = arcpy.sa.TopoContour([["vrstevnice.shp", "VYSKA"]])

# TopoToRaster - interpolace z vrstevnic
dtm = arcpy.sa.TopoToRaster(input_contours, "dtm_vinice", cell_size=10)
```

7.2.2 Krok 2: Terénní analýza

```
# Svažitost
slope = Slope(dtm, "PERCENT_RISE")

# Aspekt
aspect = Aspect(dtm)

# Vhodné svahy (< 20% + jihovýchodní až jihozápadní)
suitable_terrain = (slope < 20) & (aspect >= 135) & (aspect <= 225)
```

7.2.3 Krok 3: Analýza viditelnosti

> **Warning**
>
> Pro analýzu vrstevnic je třeba mít bodový shapefile vyhlídek. Ten v datech zatím není k dispozici. Buď si nějaký vytvořte, nebo tuto část vynechte.

```
# Předpoklad: máme vrstvu vyhlídek (body)
# Viewshed - viditelnost z vyhlídek
viewshed = Viewshed(
    dtm,
    "vyhlidky.shp",
    observer_offset=1.7  # výška pozorovatele v metrech
)

# Neviditelná místa (hodnota 0)
not_visible = viewshed == 0
```

7.2.4 Krok 4: Finální výběr

```
# Kombinace všech kritérií
final_suitable = suitable_terrain & not_visible
final_suitable.save("vhodne_pro_vinici")
```

7.3 Úloha 14: Vinice pro všechny okresy

Zadání: Řešte předchozí úlohu pro všechny okresy (každý zvlášť).

7.3.1 Struktura řešení

Zkuste sami :)

---

8 Shrnutí

8.1 Co jsme se naučili

Terénní analýzy:

- `Slope()` - výpočet svažitosti
- `Aspect()` - výpočet orientace svahu
- `Contour()` - tvorba vrstevnic

Lokální mapová algebra:

- Relační operátory: `>`, `<`, `==`, `>=`, `<=`
- Logické operátory: `&` (AND), `|` (OR), `~` (NOT)
- Funkce: `GreaterThan()`, `LessThan()`, `InList()`

Fokální mapová algebra:

- `FocalStatistics()` - operace v okolí buňky
- `NbrRectangle()`, `NbrCircle()` - definice okolí

Zonální mapová algebra:

- `ZonalStatisticsAsTable()` - statistiky pro zóny
- `TabulateArea()` - rozloha kategorií v zónách

Reklasifikace:

- `Reclassify()` - změna hodnot rastru
- `RemapRange()`, `RemapValue()` - tabulka překlasifikace

Další nástroje:

- `TopoToRaster()` - interpolace z vrstevnic
- `Viewshed()` - analýza viditelnosti

Dávkové zpracování:

- Funkce pro automatizaci analýz

- `ListRasters()` pro procházení rastrů
- Try-except pro ošetření chyb
- Export výsledků do CSV

8.2 Klíčové poznatky

1. **Rastrové objekty v paměti** - výsledky rastrových nástrojů je třeba explicitně uložit pomocí `.save()`
2. **Dva způsoby zápisu podmínek:**
   - Funkce (`GreaterThan`) - explicitní
   - Operátory (`>`) - stručné
3. **Logické operátory:**
   - Použijte `&` místo `and`
   - Použijte `|` místo `or`
   - Vždy dejte podmínky do závorek!
4. **Multikriteriální analýza** - kombinování více podmínek pomocí `&`
5. **Dávkové zpracování** - strukturujte kód do funkcí pro lepší přehlednost

---

9 Domácí úkol

9.1 Varianta A (základní)

Zadání: Analýza lyžařských svahů.

Najděte vhodná místa pro lyžařský svah:

1. Svažitost 15-35 %
2. Severní aspekt (315-45°, tj. <= 45 NEBO >= 315)
3. Nadmořská výška > 800 m

Vstup: `dtm_Liberec.tif`

Úkoly:

- Vypočítejte svažitost a aspekt
- Vytvořte masku vhodných míst

- Spočítejte celkovou rozlohu (v ha)

Odevzdejte: Skript `lyzarske_svahy.py` + screenshot výsledného rastru

9.2 Varianta B (střední)

Zadání: Analýza erozního ohrožení.

Pro každý katastr v Libereckém kraji:

1. Vypočítejte svažitost
2. Reklasifikujte svažitost do kategorií:
   - 0-3 %: 1 (žádné ohrožení)
   - 3-7 %: 2 (mírné)
   - 7-12 %: 3 (střední)
   - 12-25 %: 4 (vysoké)
   - 25 %: 5 (extrémní)
3. Pro každý katastr spočítejte rozlohu v každé kategorii
4. Vytvořte CSV s výsledky

Vstup: složka `dtm_katastry/`

Odevzdejte: Python Notebook `erozni_ohrozeni.ipynb` + CSV s výsledky

9.3 Varianta C (pokročilá)

Zadání: Komplexní analýza vhodnosti území.

Vytvořte model vhodnosti území pro fotovoltaické elektrárny:

Kritéria (každé s vahou):

1. Svažitost < 5 % (váha 0.3)
2. Jižní orientace 135-225° (váha 0.3)
3. Nadmořská výška < 500 m (váha 0.2)
4. Vzdálenost od silnic < 500 m (váha 0.2)

Postup:

1. Pro každé kritérium vytvořte rastr 0-1 (nesplňuje-splňuje)
2. Vynásobte každý rastr jeho váhou
3. Sečtěte vážené rastry $\rightarrow$ výsledek 0-1 (nevhodné-ideální)
4. Reklasifikujte výsledek do kategorií vhodnosti

Bonusově:

- Proveďte analýzu pro všechny okresy
- Vytvořte HTML report s mapami a grafy
- Vyexportujte vhodné plochy jako polygony

Odevzdejte: Notebook `fotovoltaika_analyza.ipynb` + všechny výstupy

---

10 Poznámky pro vyučujícího

10.1 Časový plán (90 min)

Čas	Obsah
0-5 min	Úvod, nastavení prostředí
5-20 min	Úlohy 1-2 (terénní analýzy)
20-40 min	Úlohy 3-5 (lokální algebra) - společně
40-50 min	Úlohy 6-7 (fokální algebra)
50-65 min	Úlohy 8-11 (zonální algebra + reklasifikace)
65-80 min	Úloha 12 v Notebooku - struktura řešení
80-85 min	Úlohy 13-14 - diskuse přístupu
85-90 min	Shrnutí, domácí úkol

10.2 Klíčové momenty

10.2.1 Úlohy 1-2 (5-20 min):

- **ZAČÍT:** Import a aktivace Spatial Analyst (všichni společně)
- **ZDŮRAZNIT:** Rastrové objekty v paměti vs. uložené na disk
- **UKÁZAT:** `.save()` metodu
- **VIZUÁLNÍ KONTROLA:** Otevřít výsledky v ArcGIS Pro

10.2.2 Úlohy 3-5 (20-40 min):

- **UKÁZAT:** Nejdřív `GreaterThan()`, pak operátory
- **ZDŮRAZNIT:** `&` místo `and`
- **DŮLEŽITÉ:** Závorky u logických operátorů
- **PRAKTICKÉ:** Interpretace výsledků (vinice, sluneční panely)

10.2.3 Úlohy 6-7 (40-50 min):

- **MOTIVACE:** Proč vyhlazovat? Ukázat na vrstevnicích
- **VIZUÁLNÍ:** Srovnání původních vs. vyhlazených vrstevnic
- **TIP:** Vyhlazení není vždy vhodné (ztráta detailů)

10.2.4 Úlohy 8-11 (50-65 min):

- **UKÁZAT:** Rozdíl mezi ZonalStatisticsAsTable a TabulateArea
- **DŮLEŽITÉ:** InList() pro vytvoření masky
- **PRAKTICKÉ:** Relativní zastoupení = průměr z masky 0/1
- **DEMO:** Reklasifikace s RemapRange

10.2.5 Úloha 12 (65-80 min):

- **PŘEPNOUT:** Z Python Window do Python Notebooku
- **UKÁZAT:** Strukturu funkce pro jeden rastr
- **ZDŮRAZNIT:** Try-except pro robustnost
- **NECHAT:** Studenty adaptovat funkci

10.2.6 Úlohy 13-14 (80-85 min):

- **DISKUTOVAT:** Postup řešení (TopoToRaster → Slope/Aspect → Viewshed)
- **UKÁZAT:** Strukturu batch processingu
- **POZNÁMKA:** Tyto úlohy jsou pro domácí práci

10.3 Rizika

1. **Spatial Analyst není aktivován**
   - Řešení: Zkontrolovat na začátku, pomoci s licencí
2. **Studenti zapomínají .save()**
   - Řešení: Opakovaně připomínat, zdůraznit rozdíl vs. vektorové nástroje
3. **Chyba: použití 'and' místo '&'**
   - Řešení: Ukázat chybovou hlášku, vysvětlit rozdíl
4. **Nedostatek času na všechny úlohy**
   - Řešení: Úlohy 13-14 pouze probrat, zbytek jako domácí úkol

5. **Komplexní úlohy (12-14) jsou náročné**
    - Řešení: Poskytnout kostru kódu, nechat studenty doplnit

10.4 Tipy

- **Vizuální kontrola:** Po každé úloze zkontrolovat výsledek v mapě
- **Data:** Mít připravená data předem, otestovat všechny skripty
- **Notebook:** Mít připravený vzorový notebook pro úlohu 12
- **TopoToRaster:** Může trvat dlouho, mít připravené výsledky
- **Viewshed:** Může být pomalý, zvážit menší testovací oblast
- **Motivace:** Zdůrazňovat praktické aplikace (vinice, lyžařské svahy, erozní ohrožení)

10.5 Běžné chyby studentů

```
# 1. Zapomínají .save()
slope = Slope("dtm")  # Není uloženo!
# Správně:
slope = Slope("dtm")
slope.save("svazitost")

# 2. Používají 'and' místo '&'
result = (slope > 5) and (aspect < 180)  # CHYBA!
# Správně:
result = (slope > 5) & (aspect < 180)

# 3. Zapomínají závorky
result = slope > 5 & aspect < 180  # CHYBA! - špatná priorita
# Správně:
result = (slope > 5) & (aspect < 180)

# 4. Zapomínají Raster() u operátorů
result = "slope" > 5  # CHYBA!
# Správně:
result = Raster("slope") > 5
```

10.6 Materiály k přípravě

☐ DTM pro Liberecký kraj
☐ Vrstva katastrů

☐ CLC 2018 rastr
☐ Složka s DTM pro katastry
☐ Data s vrstevnicemi pro úlohu 13
☐ Vzorový Python Notebook pro úlohu 12
☐ Řešení všech domácích úkolů (A, B, C)
☐ Otestovat TopoToRaster (může trvat dlouho)
☐ Otestovat Viewshed (může být pomalý)