

# Lekce 8: Textové soubory II – CSV tabulka

Python pro GIS - Profesionální práce s tabulkami

Vojtěch Barták, FŽP ČZU Praha

2025-11-24

## Table of contents

<b>1</b>	<b>Cíle lekce</b>	<b>3</b>
<b>2</b>	<b>Co je CSV formát?</b>	<b>3</b>
2.1	Comma-Separated Values . . . . .	3
2.2	Proč CSV? . . . . .	3
2.3	Problém s ručním parsováním . . . . .	4
<b>3</b>	<b>Modul csv - základy</b>	<b>4</b>
3.1	Import modulu . . . . .	4
3.2	Čtení CSV - csv.reader() . . . . .	4
3.3	Zpracování s hlavičkou . . . . .	5
<b>4</b>	<b>DictReader - práce se slovníky</b>	<b>6</b>
4.1	Problém s indexy . . . . .	6
4.2	csv.DictReader() . . . . .	6
4.3	Praktické použití . . . . .	7
<b>5</b>	<b>Zápis CSV</b>	<b>8</b>
5.1	csv.writer() . . . . .	8
5.2	csv.DictWriter() - doporučeno! . . . . .	8
<b>6</b>	<b>Praktická úloha - kompletní workflow</b>	<b>9</b>
6.1	Zadání . . . . .	9
6.2	Řešení krok za krokem . . . . .	9
6.2.1	Krok 1: Načtení dat . . . . .	9
6.2.2	Krok 2: Výpočet hustoty . . . . .	10
6.2.3	Krok 3: Filtrace . . . . .	10
6.2.4	Krok 4: Třídění pomocí math_utils . . . . .	11

6.2.5	Krok 5: Export do CSV . . . . .	12
6.3	Kompletní program . . . . .	12
<b>7</b>	<b>Propojení s GIS workflow</b>	<b>13</b>
7.1	Export atributové tabulky z ArcGIS . . . . .	13
7.2	Typický GIS workflow . . . . .	13
7.3	Příklad: Zpracování bodů zájmu . . . . .	14
<b>8</b>	<b>Praktická cvičení</b>	<b>15</b>
8.1	Cvičení 1: Statistiky okresů . . . . .	15
8.2	Cvičení 2: Agregace podle kraje . . . . .	16
<b>9</b>	<b>Shrnutí</b>	<b>17</b>
9.1	Co jsme se naučili . . . . .	17
9.2	Co bude příště? . . . . .	17
<b>10</b>	<b>Domácí úkol - příprava na test</b>	<b>17</b>
10.1	Varianta A (povinná - základní) . . . . .	17
10.2	Varianta B (příprava na test - pokročilá) . . . . .	18
10.3	Varianta C (výzva - jako na testu) . . . . .	18
<b>11</b>	<b>Cheatsheet</b>	<b>20</b>
<b>12</b>	<b>Poznámky pro vyučujícího</b>	<b>22</b>
12.1	Běžné chyby studentů . . . . .	22
12.2	Časový plán (90 min) . . . . .	22
12.3	Klíčové momenty . . . . .	23
12.3.1	DictReader vs reader (30-45 min): . . . . .	23
12.3.2	newline='' (45-60 min): . . . . .	23
12.3.3	GIS workflow (80-90 min): . . . . .	23
12.4	Rizika . . . . .	23
12.5	Tipy . . . . .	24
12.6	Příprava na test (důležité!) . . . . .	24
12.7	Materiály k přípravě . . . . .	24

# 1 Cíle lekce

Po absolvování této lekce budete umět:

- Rozumět formátu CSV a jeho problémům
- Používat modul `csv` pro čtení a zápis
- Pracovat s `csv.reader()` a `csv.DictReader()`
- Zapisovat pomocí `csv.writer()` a `csv.DictWriter()`
- Zpracovávat reálná data s hlavičkami
- Propojit CSV data s GIS workflow
- Kombinovat funkce z `math_utils` s CSV daty

Časová dotace: 90 minut

---

## 2 Co je CSV formát?

### 2.1 Comma-Separated Values

CSV = Comma-Separated Values (hodnoty oddělené čárkou)

Příklad CSV souboru:

```
Okres,Populace,Rozloha
Praha,1300000,496
Brno,380000,230
Ostrava,290000,214
```

**Struktura:** - První řádek = **hlavička** (názvy sloupců) - Další řádky = **data** - Hodnoty oddělené **čárkou** (nebo středníkem, tabulátorem...)

### 2.2 Proč CSV?

**Výhody:** - Jednoduchý textový formát - Otevřete v Excelu, LibreOffice, Google Sheets - Malá velikost souboru - Univerzální - funguje všude - **ArcGIS umí exportovat atributové tabulky jako CSV!**

**Nevýhody:** - Žádné formátování (barvy, tučné písmo...) - Jeden list (na rozdíl od Excelu)

## 2.3 Problém s ručním parsováním

Co když máte čárku v datech?

Jméno,Věk,Adresa

Jan Novák,25,Hlavní 1, Praha

Pokus o `split(",")`:

```
casti = radek.split(",")
# ['Jan Novák', '25', 'Hlavní 1', ' Praha']
# 4 části místo 3! CHYBA!
```

Řešení: Použít modul `csv`, který řeší tyto komplikace!

---

## 3 Modul csv - základy

### 3.1 Import modulu

```
import csv
```

### 3.2 Čtení CSV - `csv.reader()`

Soubor `okresy.csv`:

Okres,Populace,Rozloha

Praha,1300000,496

Brno,380000,230

Ostrava,290000,214

Čtení:

```
import csv

with open("okresy.csv", "r", encoding="utf-8") as soubor:
    reader = csv.reader(soubor)

    for radek in reader:
        print(radek)
```

Výsledek:

```
['Okres', 'Populace', 'Rozloha']
['Praha', '1300000', '496']
['Brno', '380000', '230']
['Ostrava', '290000', '214']
```

💡 `csv.reader()` vrací SEZNAMY

Každý řádek je **seznam stringů**. Čísla musíte převádět pomocí `int()` nebo `float()`.

### 3.3 Zpracování s hlavičkou

```
import csv

with open("okresy.csv", "r", encoding="utf-8") as soubor:
    reader = csv.reader(soubor)

    hlavicka = next(reader) # Přeskočit první řádek
    print(f"Sloupce: {hlavicka}")

    for radek in reader:
        okres = radek[0]
        populace = int(radek[1])
        rozloha = float(radek[2])

        hustota = populace / rozloha
        print(f"{okres}: {hustota:.1f} obyvatel/km2")
```

Výsledek:

```
Sloupce: ['Okres', 'Populace', 'Rozloha']  
Praha: 2621.0 obyvatel/km2  
Brno: 1652.2 obyvatel/km2  
Ostrava: 1355.1 obyvatel/km2
```

---

## 4 DictReader - práce se slovníky

### 4.1 Problém s indexy

```
okres = radek[0]      # Co je na indexu 0? Není jasné!  
populace = radek[1]   # Co je na indexu 1?
```

**Lepší:** Používat **názvy sloupců** místo indexů!

### 4.2 csv.DictReader()

```
import csv  
  
with open("okresy.csv", "r", encoding="utf-8") as soubor:  
    reader = csv.DictReader(soubor)  
  
    for radek in reader:  
        print(radek)
```

**Výsledek:**

```
{'Okres': 'Praha', 'Populace': '1300000', 'Rozloha': '496'}  
{'Okres': 'Brno', 'Populace': '380000', 'Rozloha': '230'}  
{'Okres': 'Ostrava', 'Populace': '290000', 'Rozloha': '214'}
```

**!** DictReader vrací SLOVNÍKY

Každý řádek je **slovník** s klíči podle hlavičky!

```
radek['Okres']      # "Praha"
radek['Populace']   # "1300000" (string!)
```

### 4.3 Praktické použití

```
import csv

okresy = []

with open("okresy.csv", "r", encoding="utf-8") as soubor:
    reader = csv.DictReader(soubor)

    for radek in reader:
        okres = {
            'nazev': radek['Okres'],
            'populace': int(radek['Populace']),
            'rozloha': float(radek['Rozloha'])
        }
        okresy.append(okres)

# Teď můžeme data zpracovávat
for o in okresy:
    hustota = o['populace'] / o['rozloha']
    print(f"{o['nazev']}: {hustota:.1f} obyvatel/km2")
```

💡 Kdy použít DictReader?

**VŽDY**, pokud má CSV hlavičku!

**Výhody:** - Čitelnější kód - Nezáleží na pořadí sloupců - Méně chyb

## 5 Zápis CSV


### 5.1 csv.writer()

```
import csv

data = [
    ['Okres', 'Populace', 'Hustota'],
    ['Praha', 1300000, 2621],
    ['Brno', 380000, 1652],
    ['Ostrava', 290000, 1355]
]

with open("vysledky.csv", "w", encoding="utf-8", newline='') as soubor:
    writer = csv.writer(soubor)

    for radek in data:
        writer.writerow(radek)
```

 **newline='' je DŮLEŽITÉ!**

Bez `newline=''` se v některých systémech (Windows) přidávají prázdné řádky:

```
with open("soubor.csv", "w", encoding="utf-8", newline='') as f:
```

**VŽDY** přidávejte `newline=''` při zápisu CSV!

### 5.2 csv.DictWriter() - doporučeno!

```
import csv

okresy = [
    {'nazev': 'Praha', 'populace': 1300000, 'hustota': 2621},
    {'nazev': 'Brno', 'populace': 380000, 'hustota': 1652},
    {'nazev': 'Ostrava', 'populace': 290000, 'hustota': 1355}
]

with open("vysledky.csv", "w", encoding="utf-8", newline='') as soubor:
    fieldnames = ['nazev', 'populace', 'hustota']
```



```
writer = csv.DictWriter(soubor, fieldnames=fieldnames)

writer.writeheader() # Zapiše hlavičku
writer.writerows(okresy) # Zapiše všechny řádky
```

Výsledný soubor:

```
nazev,populace,hustota
Praha,1300000,2621
Brno,380000,1652
Ostrava,290000,1355
```

---

## 6 Praktická úloha - kompletní workflow

### 6.1 Zadání

Máte soubor `okresy.csv`:

```
Okres,Populace,Rozloha,Kraj
Praha,1300000,496,Praha
Brno,380000,230,Jihomoravský
Ostrava,290000,214,Moravskoslezský
Plzeň,170000,261,Plzeňský
Liberec,103000,106,Liberecký
```

**Úkoly:** 1. Načíst data pomocí DictReader 2. Pro každý okres vypočítat hustotu obyvatel 3. Najít okresy s hustotou  $> 1000$  obyvatel/km<sup>2</sup> 4. Seřadit okresy podle hustoty (použijte `bubble_sort` z `math_utils`!) 5. Uložit výsledky do nového CSV

### 6.2 Řešení krok za krokem

#### 6.2.1 Krok 1: Načtení dat

```

import csv

okresy = []

with open("okresy.csv", "r", encoding="utf-8") as soubor:
    reader = csv.DictReader(soubor)

    for radek in reader:
        okres = {
            'nazev': radek['Okres'],
            'populace': int(radek['Populace']),
            'rozloha': float(radek['Rozloha']),
            'kraj': radek['Kraj']
        }
        okresy.append(okres)

print(f"Načteno {len(okresy)} okresů")

```

### 6.2.2 Krok 2: Výpočet hustoty

```

# Přidání hustoty k datům
for okres in okresy:
    okres['hustota'] = okres['populace'] / okres['rozloha']

# Výpis
for o in okresy:
    print(f"{o['nazev']}: {o['hustota']:.1f} obyvatel/km²")

```

### 6.2.3 Krok 3: Filtrace

```

huste_okresy = []

for okres in okresy:
    if okres['hustota'] > 1000:
        huste_okresy.append(okres)

print(f"\nOkresy s hustotou > 1000:")
for o in huste_okresy:
    print(f"    {o['nazev']}: {o['hustota']:.1f}")

```

#### 6.2.4 Krok 4: Třídění pomocí math\_utils

```
import math_utils

# Problém: bubble_sort třídí seznamy čísel, ne slovníky!
# Řešení: Vytvoříme seznam (hustota, okres) a seřadíme

data_k_trideni = []
for okres in okresy:
    data_k_trideni.append((okres['hustota'], okres))

# Vlastní bubble sort pro tuto úlohu
def bubble_sort_okresy(data):
    """Seřadí okresy podle hustoty (sestupně)."""
    serazene = data.copy()

    for i in range(len(serazene)):
        for j in range(len(serazene) - 1 - i):
            if serazene[j][0] < serazene[j + 1][0]: # Porovnání hustot
                serazene[j], serazene[j + 1] = serazene[j + 1], serazene[j]

    return serazene

serazene = bubble_sort_okresy(data_k_trideni)

# Extrahovat okresy
serazene_okresy = [okres for hustota, okres in serazene]
```

##### **i** Jednodušší řešení v praxi

V reálných programech byste použili:

```
okresy.sort(key=lambda x: x['hustota'], reverse=True)
```

Ale pro výukové účely je dobré ukázat, jak přizpůsobit bubble\_sort!

### 6.2.5 Krok 5: Export do CSV

```
with open("okresy_s_hustotou.csv", "w", encoding="utf-8", newline='') as soubor:
    fieldnames = ['nazev', 'populace', 'rozloha', 'kraj', 'hustota']
    writer = csv.DictWriter(soubor, fieldnames=fieldnames)

    writer.writeheader()
    writer.writerows(serazene_okresy)

print("\nVýsledky uloženy do okresy_s_hustotou.csv")
```

## 6.3 Kompletní program

```
import csv

# 1. Načtení dat
okresy = []
with open("okresy.csv", "r", encoding="utf-8") as soubor:
    reader = csv.DictReader(soubor)
    for radek in reader:
        okres = {
            'nazev': radek['Okres'],
            'populace': int(radek['Populace']),
            'rozloha': float(radek['Rozloha']),
            'kraj': radek['Kraj']
        }
        okresy.append(okres)

# 2. Výpočet hustoty
for okres in okresy:
    okres['hustota'] = okres['populace'] / okres['rozloha']

# 3. Filtrace
huste_okresy = [o for o in okresy if o['hustota'] > 1000]

# 4. Třídění (sestupně podle hustoty)
def bubble_sort_okresy(data):
    serazene = data.copy()
    for i in range(len(serazene)):
        for j in range(len(serazene) - 1 - i):
```

```

        if serazene[j]['hustota'] < serazene[j + 1]['hustota']:
            serazene[j], serazene[j + 1] = serazene[j + 1], serazene[j]
    return serazene

serazene_okresy = bubble_sort_okresy(okresy)

# 5. Export
with open("okresy_s_hustotou.csv", "w", encoding="utf-8", newline='') as soubor:
    fieldnames = ['nazev', 'populace', 'rozloha', 'kraj', 'hustota']
    writer = csv.DictWriter(soubor, fieldnames=fieldnames)
    writer.writeheader()
    writer.writerows(serazene_okresy)

# 6. Výpis výsledků
print("TOP 3 okresy podle hustoty:")
for i, okres in enumerate(serazene_okresy[:3], 1):
    print(f"{i}. {okres['nazev']}: {okres['hustota']:.1f} obyvatel/km2")

```

---

## 7 Propojení s GIS workflow

### 7.1 Export atributové tabulky z ArcGIS

V ArcGIS Pro můžete exportovat atributovou tabulku:

1. Pravý klik na vrstvu → **Open Attribute Table**
2. Menu → **Export** → **Export Table**
3. Vybrat formát: **CSV** nebo **dBASE**

**Výsledek:** CSV soubor s atributy vrstvy!

### 7.2 Typický GIS workflow

ArcGIS Pro  
(Feature class)

Export

CSV soubor

Python

Zpracování  
(výpočty,  
statistiky)

Python

Nový CSV

Import

ArcGIS Pro  
(nová vrstva)

### 7.3 Příklad: Zpracování bodů zájmu

Soubor z ArcGIS: pois.csv

```
OBJECTID,Name,Type,X,Y
1,Restaurace A,restaurant,14.4208,50.0875
2,Restaurace B,restaurant,14.4212,50.0880
3,Kavárna C,cafe,14.4198,50.0868
```

Python zpracování:

```
import csv

# Načtení
pois = []
with open("pois.csv", "r", encoding="utf-8") as f:
    reader = csv.DictReader(f)
    for row in reader:
        pois.append({
```

```

        'id': int(row['OBJECTID']),
        'name': row['Name'],
        'type': row['Type'],
        'x': float(row['X']),
        'y': float(row['Y'])
    })

# Filtrace - pouze restaurace
restaurace = [p for p in pois if p['type'] == 'restaurant']

# Export
with open("restaurace.csv", "w", encoding="utf-8", newline='') as f:
    fieldnames = ['id', 'name', 'type', 'x', 'y']
    writer = csv.DictWriter(f, fieldnames=fieldnames)
    writer.writeheader()
    writer.writerows(restaurace)

print(f"Exportováno {len(restaurace)} restaurací")

```

**Zpět do ArcGIS:** 1. **Add Data** → vybrat `restaurace.csv` 2. **Display XY Data** → nastavit X, Y sloupce 3. **Export Features** → uložit jako shapefile/geodatabase

#### 💡 Výhoda tohoto přístupu

- Rychlé zpracování velkých dat
- Opakovatelné (skript můžete spustit znovu)
- Kombinuje sílu Pythonu s vizualizací ArcGIS
- Můžete použít vlastní funkce (`math_utils!`)

## 8 Praktická cvičení

### 8.1 Cvičení 1: Statistiky okresů

Použijte soubor `okresy.csv` a vytvořte program, který:

1. Načte data pomocí `DictReader`
2. Vypočítá pro každý okres:
  - Hustotu obyvatel

- Zda je “velký” (populace > 200 000)
3. Spočítá statistiky:
    - Průměrnou hustotu
    - Počet velkých okresů
    - Celkovou populaci
  4. Uloží výsledky do `statistiky_okresu.csv`:

```
nazev,populace,rozloha,hustota,velky
Praha,1300000,496,2621.0,True
Brno,380000,230,1652.2,True
...
```

## 8.2 Cvičení 2: Agregace podle kraje

Vytvořte program, který:

1. Načte `okresy.csv`
2. Seskupí okresy podle kraje
3. Pro každý kraj vypočítá:
  - Celkovou populaci
  - Celkovou rozlohu
  - Průměrnou hustotu
4. Uloží do `kraje_agregace.csv`:

```
Kraj,Celkova_populace,Celkova_rozloha,Prumerna_hustota
Praha,1300000,496,2621.0
Jihomoravský,380000,230,1652.2
...
```

Nápověda pro agregaci:

```
kraje = {}
for okres in okresy:
    kraj = okres['kraj']
    if kraj not in kraje:
        kraje[kraj] = []
    kraje[kraj].append(okres)
```



## 9 Shrnutí

### 9.1 Co jsme se naučili

**CSV formát** - struktura, výhody, problémy  
**csv.reader()** - čtení CSV jako seznamy  
**csv.DictReader()** - čtení CSV jako slovníky (doporučeno!)  
**csv.writer()** - zápis CSV  
**csv.DictWriter()** - zápis CSV ze slovníků (doporučeno!)  
**newline=''** - důležité pro správný zápis  
**GIS workflow** - export z ArcGIS → Python → import zpět  
**Propojení s funkcemi** - kombinace CSV s math\_utils

### 9.2 Co bude přístě?

**Příští týden: TEST!**

Test bude obsahovat: - **Písemná část** - pochopení kódu, doplňování - **Praktická část** - zpracování CSV souboru s použitím funkcí

**Za 2 týdny: OOP (Objektově orientované programování)** - Třída Pes - Propojení s ArcPy objekty - Příprava na práci s geometriemi

---

## 10 Domácí úkol - příprava na test

### 10.1 Varianta A (povinná - základní)

Vytvořte soubor `teploty_mesic.csv`:

```
Den,Teplota,Srazky
1,15,0
2,18,5
3,22,0
4,19,12
5,16,8
```

**Napište program, který:** 1. Načte data pomocí DictReader 2. Vypočítá: - Průměrnou teplotu  
- Celkové srážky - Den s nejvyšší teplotou 3. Uloží statistiky do `vysledky_pocasi.txt`:

Statistiky počasí  
=====  
Průměrná teplota: 18.0°C  
Celkové srážky: 25 mm  
Nejteplejší den: 3 (22°C)

## 10.2 Varianta B (příprava na test - pokročilá)

Vytvořte modul `csv_utils.py` s funkcemi:

```
def nacti_csv_dict(nazev_souboru):  
    """Načte CSV jako seznam slovníků."""  
    # ...  
  
def uloz_csv_dict(nazev_souboru, data, fieldnames):  
    """Uloží seznam slovníků jako CSV."""  
    # ...  
  
def vypocitaj_statistiky(data, sloupec):  
    """Vypočítá průměr, min, max pro daný sloupec.  
  
    Returns:  
        dict: {'prumer': ..., 'minimum': ..., 'maximum': ...}  
    """  
    # ...
```

Použijte tyto funkce v hlavním programu!

## 10.3 Varianta C (výzva - jako na testu)

Máte `zakaznici.csv`:

```
ID,Jmeno,Vek,Utrata  
1,Jan Novák,25,1500  
2,Marie Svobodová,32,2300  
3,Petr Dvořák,28,1800  
4,Jana Nováková,45,3200
```

**Napište program, který:** 1. Načte data 2. Vypočítá průměrnou útratu 3. Najde zákazníky s útratou nad průměr 4. Seřadí je podle útraty (použijte vlastní třídící funkci!) 5. Uloží do

`vip_zakaznici.csv` s přidaným sloupцем `Kategorie`: - “VIP” pokud útrata  $> 2 \times$  průměr - “Premium” pokud útrata  $>$  průměr - “Standard” jinak

**Struktura programu:** - `zakaznici_utils.py` - funkce pro zpracování - `main.py` - hlavní program

---

## 11 Cheatsheet

```
import csv

# === ČTENÍ CSV ===
# S csv.reader() - seznam
with open("data.csv", "r", encoding="utf-8") as f:
    reader = csv.reader(f)
    hlavicka = next(reader) # Přeskočit hlavičku
    for radek in reader:
        # radek je seznam ['Praha', '1300000', ...]

# S DictReader - slovník (DOPORUČENO!)
with open("data.csv", "r", encoding="utf-8") as f:
    reader = csv.DictReader(f)
    for radek in reader:
        # radek je slovník {'Okres': 'Praha', ...}
        nazev = radek['Okres']
        populace = int(radek['Populace'])

# === ZÁPIS CSV ===
# S csv.writer()
with open("vysledky.csv", "w", encoding="utf-8", newline='') as f:
    writer = csv.writer(f)
    writer.writerow(['Sloupec1', 'Sloupec2']) # Hlavička
    writer.writerow(['hodnota1', 'hodnota2']) # Data

# S DictWriter() - DOPORUČENO!
data = [
    {'nazev': 'Praha', 'hodnota': 100},
    {'nazev': 'Brno', 'hodnota': 200}
]
with open("vysledky.csv", "w", encoding="utf-8", newline='') as f:
    fieldnames = ['nazev', 'hodnota']
    writer = csv.DictWriter(f, fieldnames=fieldnames)
    writer.writeheader() # Zapiše hlavičku
    writer.writerows(data) # Zapiše všechny řádky

# === PŘEVODY ===
radek['Populace'] # "1300000" (string!)
int(radek['Populace']) # 1300000 (int)
```

```
float(radek['Rozloha'])          # 496.0 (float)

# === DŮLEŽITÉ ===
# VŽDY: encoding="utf-8"
# VŽDY při zápisu: newline=''
# DictReader/DictWriter > reader/writer
```

---

## 12 Poznámky pro vyučujícího

### 12.1 Běžné chyby studentů

```
# 1. Zapomínají newline=''
with open("data.csv", "w", encoding="utf-8") as f: # Chyba - prázdné řádky!
# Správně:
with open("data.csv", "w", encoding="utf-8", newline='') as f:

# 2. Nepoužívají DictReader
reader = csv.reader(f)
for radek in reader:
    okres = radek[0] # Co je na indexu 0? Není jasné!
# Správně:
reader = csv.DictReader(f)
for radek in reader:
    okres = radek['Okres'] # Jasné!

# 3. Zapomínají převést na číslo
populace = radek['Populace'] # "1300000" (string!)
hustota = populace / rozloha # CHYBA!
# Správně:
populace = int(radek['Populace'])

# 4. Chybný fieldnames
fieldnames = ['nazev', 'populace']
writer = csv.DictWriter(f, fieldnames=fieldnames)
writer.writerow({'jmeno': 'Praha', 'pocet': 1000}) # CHYBA - neexistující klíče!

# 5. Zapomínají writeheader()
writer = csv.DictWriter(f, fieldnames=fieldnames)
writer.writerow(data) # Chybí hlavička!
# Správně:
writer.writeheader()
writer.writerow(data)
```

### 12.2 Časový plán (90 min)

Čas	Obsah
0-15 min	CSV formát, problémy, modul csv
15-30 min	csv.reader(), zpracování
30-45 min	DictReader - lepší způsob
45-60 min	Zápis - writer, DictWriter
60-80 min	Praktická úloha - kompletní workflow
80-90 min	GIS propojení, příprava na test

## 12.3 Klíčové momenty

### 12.3.1 DictReader vs reader (30-45 min):

- **DŮLEŽITÉ:** Zdůraznit, že DictReader je lepší
- Ukázat rozdíl na projektoru vedle sebe
- Říct: “V praxi používejte vždy DictReader pokud máte hlavičku”

### 12.3.2 newline='' (45-60 min):

- Vysvětlit proč je potřeba
- Ukázat příklad ŠPATNÉHO výstupu bez newline
- Říct: “Zapamatujte si: při zápisu CSV = vždy newline=’ ’”

### 12.3.3 GIS workflow (80-90 min):

- **Motivace pro test příští týden**
- Ukázat diagram: Export → Python → Import
- Zdůraznit: “Tohle budete dělat v ArcPy!”
- Říct: “Test bude podobný - CSV + funkce + výpočty”

## 12.4 Rizika

### 1. DictReader může být matoucí (20 min místo 15)

- Řešení: Praktické ukázky, porovnání s reader()
- Nechat studenty experimentovat

### 2. Třídění slovníků je složité (Krok 4)

- Řešení: Ukázat jednodušší verzi se `sort(key=...)`
- Říct: “Na testu nebude třídění slovníků, jen seznamů čísel”

- Alternativně přeskočit a dát jako bonusový úkol

### 3. Studenti nestihnou celou úlohu

- Řešení: Kroky 1-3 společně, 4-5 jako domácí úkol
- Mít připravené řešení na sdílení

## 12.5 Tipy

- **Vytvořte okresy.csv** předem a sdílejte se studenty
- Ukažte **Excel** vedle Pythonu - ať vidí, že jde o stejná data
- Zdůrazněte **DictReader = vždy když máte hlavičku**
- **GIS propojení:** “V ArcPy budete exportovat tabulky, zpracovávat, importovat zpět - přesně tohle!”
- Před koncem připomeňte: “**Příští týden TEST!**”

## 12.6 Příprava na test (důležité!)

**Co bude na testu:** - Čtení CSV pomocí DictReader - Zpracování dat (výpočty, filtrace) - Použití funkcí (vlastních nebo z modulu) - Zápis výsledků do nového CSV - Pochopení kódu (co dělá tento program?)

**Co NEBUDE:** - Složité algoritmy (třídění slovníků) - OOP (to až po testu) - ArcPy (to až později)

**Doporučení studentům:** - Projděte si všechny domácí úkoly - Zopakujte funkce (math\_utils) - Procvičte DictReader a DictWriter - Zkuste si vytvořit vlastní CSV a zpracovat ho

## 12.7 Materiály k přípravě

- ☐ Soubor `okresy.csv` s ukázkovými daty
- ☐ Soubor `teploty_mesic.csv` pro domácí úkol
- ☐ Řešení všech cvičení
- ☐ **Ukázkový test** - připravit pro příští týden!
- ☐ Diagram GIS workflow (export → Python → import)