Übung 3: Shapefile-Header, Byte für Byte

Geoinformatik Sommersemester 2024 – Güren Tan Dinga

Abgabe:

Die Übung ist als `*.zip`-Datei via Moodle oder als Teil eines Repositories einzureichen. Das späteste Abgabedatum ist

Montag, der 20.05.2021, 13:00 Uhr.

Der Dateiname muss dabei euren Vor- und Nachnamen enthalten:

vorname_nachname.zip (z.B.: gueren_dinga.zip)

Übungsbeschreibung:

siehe data Ordner

Gegeben ist eine Shape-Datei `ne_10m_populated_places.shp`. Die Aufgabe besteht darin, den Header der Shape-Datei zu parsen. Der Shapefile-Header enthält neben Dateieigenen Informationen beispielsweise Informationen zur Ausdehnung des Datensatzes. Informationen zum Shape-Datenformat finden Sie im von ESRI angebotenen Whitepaper unter https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf oder in der beigelegten PDF-Datei `shapefile.pdf`.

Zum Parsen des Shapefile-Headers empfiehlt sich die Python-Bibliothek `struct`, genauer die Funktion `unpack`: https://docs.python.org/3/library/struct.html#struct.unpack

Die Funktion erwartet zwei Parameter: ein Byte-Objekt und ein Format-String, welcher angibt, wie jenes Byte-Objekt interpretiert werden soll. Der Format-String gibt sowohl Aufschluss darüber, wie die Bytes im Speicher abgelegt sind (Little- oder Big-Endian, s. https://docs.python.org/3/library/struct.html#byte-order-size-and-alignment), als auch darüber, welcher Datentyp das Byte-Objekt repräsentiert (https://docs.python.org/3/library/struct.html#format-characters). Diese zahlreichen Datentypen erlauben eine effiziente Nutzung von Bytes um spezifische Werte darzustellen.

Im Rahmen dieser Übung lesen wir eine Shape-Datei, zu Übungszwecken, Stück für Stück ein. In der Regel werden im Sinne der Effizienz größere Teile von Dateien eingeladen.

Beispiel: Byte Unpacking

```
from struct import pack, unpack
                  float=f, i=int
x = pack("<fi", 23.25, 3240)
y = unpack("<fi", x)
print(x, y)

>>> b'\x00\x00\xbaA\xa8\x0c\x00\x00' (23.25, 3240)
```

Mit der Funktion `pack` haben wir die Möglichkeit unsere Gleitkommazahl und unsere Ganzzahl als bytes darzustellen. Der variablen y wird die Funktion `unpack` zugewiesen. Das erste Argument sagt in diesem Falle aus, dass es sich um ein little endian (signalisiert durch `<`), eine Gleitkommazahl (signalisiert durch `f`) und eine Ganzzahl (signalisiert durch `i`) handelt. Das zweite Argument, `x`, stellt unsere byte-Objekt dar.

Wir können auch lediglich den ersten Wert *unpacken*. Aus der Python-Dokumentation zu `struct` wissen wir, dass für den Datentyp *float* 4 bytes reserviert sind. Entsprechend lesen wir die ersten vier Elemente von `x` anstelle des gesamten Wertebereichs:

```
from struct import pack

x = pack("<fi", 23.25, 3240)
y = unpack("<f", x[:4])[0]
print(y)    [:4]entpackt die ersten vier Bytes aus x als float,
            [0]=
>>> 23.25
```

Darüber hinaus kommt es vor, dass wir eine beliebige byte-Menge aus einer Datei lesen möchten. Dies ermöglicht read:

```
with open(txt_path, "rb") as txt:
    # txt enthält "hello, world!"
    x = txt.read(5).decode("utf-8")
    print(x)

>>> hello
```

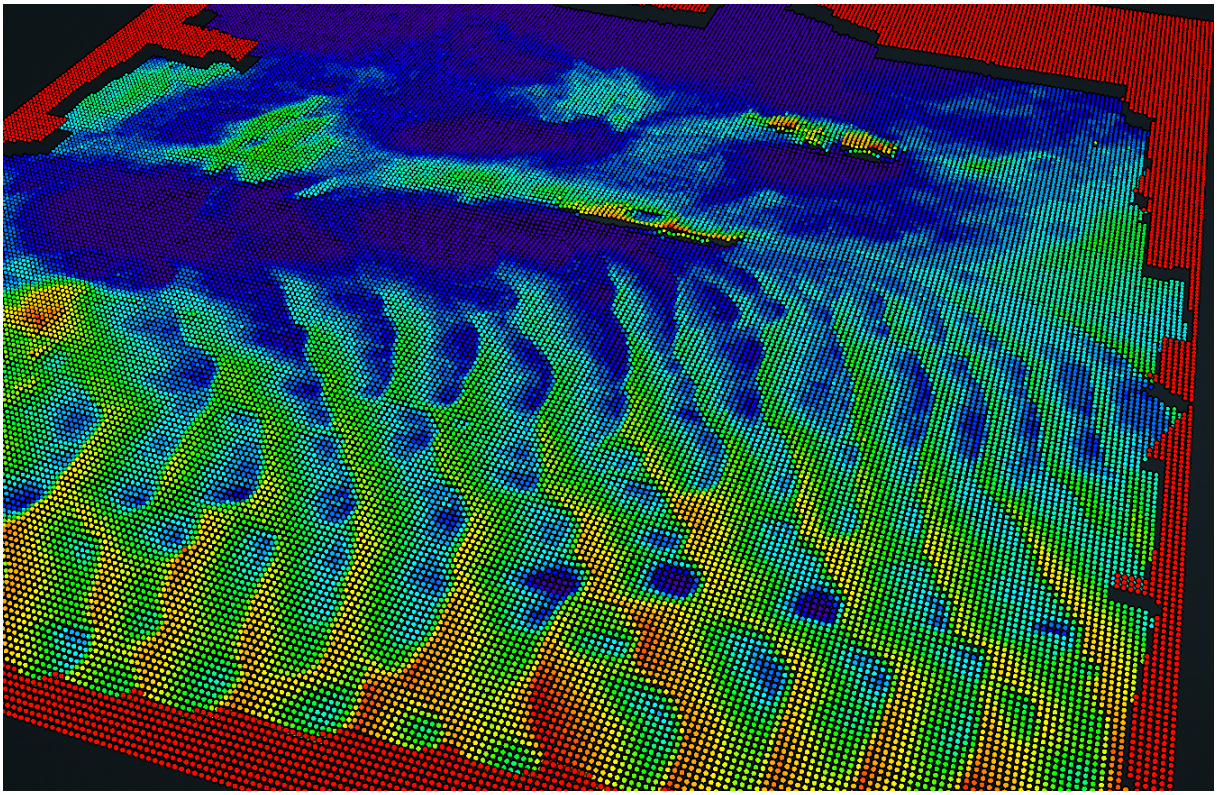
Aufgabe 01: Shape File Header

- Erzeugen Sie eine Funktion mit dem Namen `shape_header`. Dieser Funktion soll der Pfad zu einer Shape-Datei übergeben werden können.
- Erwartet ist die Ausgabe des filecodes, der filelength, der version, dem shapetype und der bounding box der Shape Datei. Die Ausgabe soll formatiert werden und kann beispielsweise so aussehen:

```
Header of data/ne_10m_populated_places.shp:
Filecode        = 9994
File Lenght     = 102838
Version         = 1000
Shape Type      = 1
Bounding Box    = [-179.5899789, -89.9999998, 179.3833036, 82.4833232]
Zmin, Zmax      = (0.0, 0.0)
Mmin, Mmax      = (0.0, 0.0)
```

Aufgabe 02: las Header

Das LAS-Format (von „Laser“) stellt ein Dateiformat zur Speicherung von LiDAR-Punktwolken dar. LAS ist ein von der APRS (American Society for Photogrammetry and Remote Sensing) spezifizierter Standard, offen und binär. Die nachfolgende Abbildung zeigt eine Sonar-Aufnahme der Elbe mit zwei Wracks (s. https://pointclouds.hcu-hamburg.de/ - Wracks).



Auch die ASPRS stellt unter dem nachfolgenden Link (alternativ in den Übungsunterlagen als `LAS.pdf`) https://www.asprs.org/wp-content/uploads/2019/07/LAS_1_4_r15.pdf eine Format-Definition für das LAS-Format zur Verfügung. Die Spezifikation umfasst alle Informationen um folgende Aufgabe zu lösen:

- Entwickeln Sie eine Funktion las_header. Dieser Funktion soll der Pfad zu einer LAS oder LAZ (kompressiertes LAS-Format) übergeben werden können.
- Erwartet ist die Ausgabe der Version der Datei (im Format MAJOR.MINOR, bspw. 1.0) sowie das Datum der Erzeugung der LAS-Datei (im Format DD.MM.YYYY).