

Programm zur Analyse meiner Kontoauszüge bei Easybank

Kontoanalyse.py

usage: kontoanalyse.py [-h] [--export CSV] [--suche BEGRIFF]
 [--feld {konto,name,referenz,alle}] [--regex]
 [--von DATUM] [--bis DATUM] [--datum {buchung,umsatz}]
 datei

Kontoauszug-CSV nach Empfängern auswerten.

positional arguments:

datei Pfad zur CSV-Datei

options:

-h, --help show this help message and exit
--export CSV Ergebnis (bzw. Treffer) als CSV speichern
--suche BEGRIFF Buchungen suchen statt Kategorien auszuwerten
--feld {konto,name,referenz,alle}
 Suchfeld: konto, name, referenz oder alle (Standard)
--regex Suchbegriff als regulären Ausdruck interpretieren
--von DATUM Suchzeitraum ab (inklusive), TT.MM.JJJJ oder JJJJ-MM-
 TT
--bis DATUM Suchzeitraum bis (inklusive), TT.MM.JJJJ oder JJJJ-MM-
 TT
--datum {buchung,umsatz}
 Welches Datum für --von/--bis gilt: buchung (Standard)
 oder umsatz

Beispiele:

```
python kontoanalyse.py x.csv  
python kontoanalyse.py x.csv --suche obi --von 01.05.2026 --bis 31.05.2026  
python kontoanalyse.py x.csv --suche miete --feld referenz --von 01.01.2026  
python kontoanalyse.py x.csv --suche "obi|diskont" --feld name --regex --bis 2026-05-12 --datum  
umsatz
```

Der Prompt

Hier ist ein Prompt, der das Ergebnis in einem Durchgang beschreibt. Er enthält die Entscheidungen, die sich erst im Lauf des Gesprächs ergeben haben, etwa die Referenz-Erkennung und dass „Gutschrift“ keine Rückbuchung ist.

Schreibe ein Python-3-Kommandozeilenprogramm `kontoanalyse.py` (eine Datei, nur Standardbibliothek, Beträge mit Decimal), das einen Kontoauszug-CSV-Export auswertet.

Teste es anschließend mit den Beispielzeilen unten und liefere die Datei aus.

Eingabeformat

- Semikolon-getrennt, keine Kopfzeile (falls doch eine da ist: automatisch überspringen),

- CRLF, Dezimalkomma, Tausenderpunkt. Encoding: erst utf-8-sig, dann cp1252.

- Spalten: Eigene IBAN; Buchungstext; Buchungsdatum; Umsatzdatum (Valuta); Betrag; Währung.

- Daten im Format TT.MM.JJJJ. Zeilen mit zu wenigen Spalten oder ungültigem Betrag

- mit Warnung auf stderr überspringen. Bei mehreren Währungen warnen, nicht umrechnen.

- Beispielzeilen (Kartenzahlungen, Backslashes wörtlich):

- AT4123420020011944460;Bezahlung Karte

- MC/000004612|OEAMTC 3145 K002 13.05. 17:42|OEAMTC BADEN 1700\\BADEN\\2512;13.05.2026;13.05.2026;-146,90;EUR

- AT41142320020011944460;Bezahlung Karte

- MC/000004611|POS 0724 K002 13.05. 10:10|OBI SAGT DANKE\\BADEN\\2500;13.05.2026;13.05.2026;-24,82;EUR

- AT412320020011944460;Bezahlung Karte

- MC/000004610|POS OAT3 K002 12.05. 10:50|DISKONT 8007\\BADEN\\2500;13.05.2026;12.05.2026;-49,21;EUR

- Überweisungen enthalten im Buchungstext z. B.

- "BAWAATWXXX AT78142002230012235861 Kalr Maxinger" (optionale BIC, IBAN ohne

- Leerzeichen, danach der Name).

Kategorien (Standardmodus ohne --suche)

Jede Buchung bekommt eine Kategorie, je Kategorie werden Anzahl und Summe gebildet.

1. Rückbuchungen: Enthält der Buchungstext (ohne Beachtung der Groß-/Kleinschreibung)

- Rückbuchung (auch Rueckbuchung/Ruckbuchung), Storno, Retoure, Rückerstattung,

- Rückzahlung, Rücküberweisung, Chargeback, Refund oder Reversal,

ist die Kategorie

"RÜCKBUCHUNGEN". Sie werden nicht mit den Ursprungsbuchungen verrechnet.

"Gutschrift" ist bewusst KEIN Schlüsselwort. Die Liste soll als Konstante leicht änderbar sein.

2. Sonst Empfänger = Kontobezeichnung: Steht im Text eine fremde IBAN (nicht die eigene),

ist die Kategorie der Name hinter der IBAN (Originalschreibweise, z. B.

"Kalr Maxr"). Der Name reicht bis zum Textende oder bis zu "|" bzw. "\".

Eine beschriftete Referenz im Namen ("Verwendungszweck: ...") wird abgeschnitten.

Ist zu einer IBAN irgendwo in der Datei ein Name bekannt, wird er auch für Buchungen

derselben IBAN ohne Namen verwendet (IBAN->Name-Verzeichnis über die ganze Datei).

Nur wenn nirgends ein Name vorkommt, ist die IBAN selbst die Kategorie.

3. Kartenzahlung (Text mit "|"): Händlername = letztes Segment nach "|", bis zum ersten

Backslash, nachgestellte Zahlen (PLZ/Filialnr.) entfernt, in Großbuchstaben

(z. B. "OEAMTC BADEN", "OBI SAGT DANKE", "DISKONT").

4. Fallback: bereinigter Text, 60 Zeichen, Großbuchstaben.

Ausgabe: Tabelle Kategorie / Anzahl / Summe (deutsches Zahlenformat 1.234,56 plus

Währung), aufsteigend nach Summe sortiert (größte Ausgabe zuerst), darunter GESAMT.

`--export CSV` schreibt dieselbe Tabelle (Semikolon, utf-8-sig).

Zerlegung jeder Buchung für die Suche

- Konto: fremde IBAN aus dem Text (sonst leer)

- Zielname: wie die Kategorie unter Punkt 2/3, aber unabhängig von "RÜCKBUCHUNGEN"

- Zahlungsreferenz/Überweisungszweck, in dieser Reihenfolge: (a) beschriftet mit

Zahlungsreferenz|Verwendungszweck|Referenz|Zahlungsgrund|Zweck|Ref, gefolgt von ":" oder

"=" (bis "|", "\" oder Zeilenende); (b) bei Überweisungen der Text nach dem Namen

hinter "|" oder "\"; (c) bei Kartenzahlungen die Belegnummer (z. B. MC/000004612)

Suche (--suche BEGRIFF)

- `--feld konto|name|referenz|alle`` (Standard `alle``). Die Suche ist ein Teilstring-Vergleich ohne Beachtung der Groß-/Kleinschreibung. Konto-Suche ignoriert Leerzeichen im Begriff.
Ein Name wird nicht im Feld Konto gefunden und umgekehrt.
- `--regex``: Begriff ist ein regulärer Ausdruck (`re.search`, `IGNORECASE`). Die Konto ist ohne Leerzeichen gespeichert. Ungültiger Ausdruck -> verständliche Fehlermeldung, Exit-Code 1. Ohne `--regex` bleibt die Suche wörtlich.
- Zeitraum: `--von`` und `--bis`` (TT.MM.JJJJ oder JJJJ-MM-TT, jeweils optional, beide Grenzen inklusive), `--datum buchung|umsatz`` (Standard `buchung = Buchungsdatum`, `umsatz = Umsatzdatum` aus Spalte 4). Buchungen ohne gültiges Datum fallen bei gesetztem Zeitraum raus. Ungültiges Datum oder `von > bis` -> Fehler.
- `--feld`, `--regex`, `--von`, `--bis` sind nur zusammen mit `--suche` erlaubt (`argparse`-Fehler).
- Die IBAN->Name-Zuordnung wird immer über ALLE Buchungen gebildet, die Filter wirken erst danach.
- Ausgabe: bei Zeitraum eine Zeile "Suche '...' (Feld: ...) | Zeitraum: ... (Buchungsdatum|Umsatzdatum)". Dann Trefferliste mit Spalten Buchung, Umsatz, Betrag (rechtsbündig), Konto, Zielname, Referenz, Kategorie (lange Werte mit "... kürzen, leere Werte als "-"), darunter "N Treffer". Danach "Kategorien der Treffer:" mit Tabelle Kategorie/Anzahl/Summe und GESAMT (Rückbuchungen bleiben eigene Kategorie). Keine Treffer -> Meldung (mit Zeitraum, falls gesetzt).
- `--export CSV`` speichert bei der Suche die Treffer (Semikolon, utf-8-sig, Spalten:

Buchungsdatum;Valuta;Betrag;Waehrung;Konto;Zielname;Referenz;Kategorie;Buchungstext).

Hilfetext

`argparse` mit `RawDescriptionHelpFormatter` und diesem Epilog

"Beispiele:" (wörtlich):

```
python kontoanalyse.py x.csv --suche obi --von 01.05.2026 --bis 31.05.2026
```

```
python kontoanalyse.py x.csv --suche miete --feld referenz --von 01.01.2026
```

```
python kontoanalyse.py x.csv --suche "obi|diskont" --feld name
--regex --bis 2026-05-12 --datum umsatz
```

Ein Modul-Docstring am Dateianfang erklärt Format, Kategorien, Suche und Aufrufe.

Test

Prüfe mit den drei Beispielzeilen (Summe -220,93 EUR) plus selbst ergänzten Zeilen:

Überweisung mit BIC+IBAN+Name und "|Miete Mai 2026", eine mit "Verwendungszweck: ...", eine nur mit IBAN (muss dem bekannten Namen zugeordnet werden), eine Storno-Buchung.

Teste alle Suchfelder, --regex, den Zeitraum (inkl. --datum umsatz) und Fehlerfälle.

Hinweise zur Verwendung

Du kannst mit den Beispielzeilen aus deiner Datei weiterarbeiten. Für Überweisungen sind es nur meine Annahmen, echte Zeilen deiner Bank würden die Referenz-Erkennung verbessern.

Wenn du das Ergebnis in einem neuen Chat prüfst, achte auf zwei Stellen, die leicht abweichen können: die Rückbuchungs-Schlüsselwörter und die Händlernamen bei Kartenzahlungen.