Übung: Unix
Einführung: Aufgabe 3

Malte Heins
Helga Karafiat

- Themen
  - Parameterverarbeitung
  - Variablen
  - Schleifen
  - bedingte Anweisungen
  - Ganzzahlarithmetik

- Aufgabenstellung
  - Kleiner Ganzzahl-Taschenrechner in Postfixnotation

Taschenrechner

- Ort: **ueb03**
- Name: **pfcalc.sh**
- Durchführen von einfachen mathematischen Berechnungen
  - Addition (ADD)
  - Subtraktion (SUB)
  - Multiplikation (MUL)
  - Ganzzahlige Division (DIV)
  - Rest der ganzzahligen Division (MOD)
  - Exponentiation (EXP)
- Schreiben auf **stdout** und **stderr**
- Verarbeitung beliebig vieler Kommandozeilenparameter

Taschenrechner - Ausgaben

- Im Erfolgsfall:
  - Ausgabe des Berechnungsergebnisses auf `stdout`
  - Ausgabe der Zwischenberechnungen in Präfix-Notation auf `stderr`
- Im Fehlerfall:
  - Ausgabe einer aussagekräftigen einzeiligen Fehlermeldung mit dem Präfix „ERROR: “ auf `stderr`
  - Danach Ausgabe der usage auf `stderr` (ohne Leerzeile dazwischen)

```
./pfcalc.sh 1 2 ADD 2 SUB 7 MUL 2 EXP

> ADD 1 2
> SUB 3 2
> MUL 1 7
> EXP 7 2
49
```

Legende
Rot: stderr
Blau: stdout

- Reihenfolge von `stderr` und `stdout` bei der Ausgabe ist nicht vorgeschrieben (kann also auch vertauscht sein)
- Reihenfolge der Ausgaben in der History ergibt sich aus den angegebenen Berechnungen, ist also eindeutig

- Exitcode > 0
- Ausgabe: Aussagekräftige Fehlermeldung auf stderr (Präfix ERROR:), gefolgt vom Hilfetext auf stderr
- Mögliche Fehlerfälle (Beispiele):
  - Unbekannte Operatoren
  - Falsche Aufrufsyntax
  - Division durch Null
  - Exponent negativ
- Keine Überprüfung nötig, ob die Zahlen gültig sind
- Keine Überlaufprüfung bei der Berechnung

Anforderungen

- Modularisierung
  - Mindestens Funktion zur Hilfeausgabe
    (Hilfe ist Bestandteil des Programms – keine Auslagerung)
  - Ggfs weitere Funktionen zur sinnvollen Kapselung
    (z.B. Erzeugung der History, Berechnung der Exponentiation)
- Umfang
  - Aufgabe in < 150 Zeilen lösbar (ohne Kommentare und Leerzeilen)
  - Lösungen mit mehr als 250 Zeilen werden nicht akzeptiert
- Sinnvolle Kommentierung und Formatierung erforderlich
- Anforderungen aus Aufgabe 1 (Dokumentation, Hinweise zu Skripten etc.) gelten in jeder Aufgabe

sh vs. bash

- Verwendung der bash bei dieser Aufgabe ausdrücklich untersagt
- Skript soll möglichst POSIX-konform und portabel sein
  (Ausnahme Schlüsselwort: `local`)

Hilfe zum eigenständigen Testen

- Vorgabe Beispieltest
  - Testen der korrekten Hilfeausgabe
  - Testen des Returncodes
  - Testen von Ausgaben
- Bei dieser Aufgabe weiterhin zwei sinnvolle Testarten (siehe beispieltest.arnold)
  - Test der Standardausgabe (direkt in Arnold)
  - Vergleich von Ausgabedatei mit erwarteter Datei (mit Hilfe von **`diff`**)

    Hinweis: **`echo "Hallo" | diff - hallo.exp`**
    bewirkt, dass `diff` von `stdin` liest, also „Hallo“ mit dem Inhalt von hallo.exp vergleicht
- Sinnvolles Testvorgehen
  - Ausnutzung von bedingter Ausführung im Fehler- bzw Erfolgsfall (**&&, ||**)
  - Nutzung von Umleitungsoperatoren
    - getrennte Behandlung von `stdout` und `stderr`
    - Verwerfen von stdout oder `stderr`

Testen und Abnahme

- Servertest
  - Verfügbar nach Ankündigung in der Newsgroup
  - Achtet darauf euer Skript rechtzeitig vor dem Servertest einzuchecken
  - Servertestergebnis bestimmt die Bewertung der Aufgabe
- Eigenes Testen
  - Eigener automatisierter Test gefordert
- Abnahme
  - Erklären des Skriptes
  - Erklären der eigenen Tests