ESP/Ass2 WS17

Aus Progwiki
< ESP

Inhaltsverzeichnis

Textadventure

Im Rahmen des zweiten Übungsbeispiels soll ein einfaches Computerspiel (ein multiple choice adventure - ähnlich einem Spielbuch) erstellt werden. Hierbei handelt es sich um eine geschriebene Geschichte auf deren Verlauf der Leser durch Auswahl aus mehreren Entscheidungen Einfluss nehmen kann.

Die Geschichte besteht aus einzelnen Elementen. Jedes dieser Elemente endet mit einer Auswahl (Wahl A oder Wahl B), mit welcher der Spieler entscheidet, wie die Geschichte weitergeht. Es gibt also im Endeffekt verschiedene mögliche Enden, welche der Spieler erreichen kann.

Funktionsumfang

Kommandozeilenparameter

Der Dateiname für die Startdatei wird als Kommandozeilenparameter übergeben.

Um das Programm aufzurufen, wird der folgende Befehl verwendet:

```
./ass2 [filename]
```

[filename] ist hierbei der Dateiname des Startelements. Falsche Eingaben bzw. Files sollen wie unter Rückgabewerte und Fehlermeldungen beschrieben behandelt werden.

Beispiel

```
./ass2 start_of_story.txt
```

Anmerkung: Die Dateiendung *.txt muss nicht vorhanden sein, damit eine Datei gültig ist.

Dateiformat

Die Elemente des Spiels werden als einzelne Textdateien gespeichert. Jede Textdatei ist wie folgt aufgebaut:

```
[Titel des Elements]\n
[Dateiname Wahl A]\n
[Dateiname Wahl B]\n
[Anzeigetext]
```

In der ersten Zeile wird der Titel des Elements gespeichert (siehe Datenstruktur und Ausgabe). Die zweite und dritte Zeile beinhalten jeweils einen Dateinamen einer Datei, in der das nächste Element für die jeweilige Auswahlmöglichkeit gefunden werden kann. Sind im aktuellen Element keine weiteren Auswahlmöglichkeiten vorhanden (weil man das Ende der Geschichte erreicht hat), so bestehen **beide** diese Zeilen jeweils aus:

```
-\n
```

Es ist kein Fall vorgesehen, bei dem nur eine der Auswahlmöglichkeiten für eine weitere Datei genutzt wird (und die andere einen Endpunkt darstellt).

Ab der vierten Zeile folgt der Text des aktuellen Elements (welcher wieder selbst aus beliebig vielen Zeilen bestehen darf).

Jede Abweichung von dieser Struktur bedeutet ein korruptes File (siehe Fehlermeldungen).

Beispieldatei

```
Kapitel 1
chapter_21.txt
chapter_42.txt
'Would you tell me, please, which way I ought to go from here?'
'That depends a good deal on where you want to get to,' said the Cat.
'I don't much care where -' said Alice.
'Then it doesn't matter which way you go,' said the Cat.
'- so long as I get SOMEWHERE,' Alice added as an explanation.
'Oh, you're sure to do that,' said the Cat, 'if you only walk long enough.'
```

Datenstruktur

Entwerfen Sie eine Struktur, um die einzelnen Elemente der Geschichte speichern zu können. Diese Elemente müssen folgende Inhalte abdecken können:

- einen C-String für den Titel
- einen C-String für den Text
- einen Verweis auf das Element für Wahl A
- einen Verweis auf das Element für Wahl B

Kreisverweise

In der Grundversion, welche Sie für die Abgabe implementieren müssen, können zwischen den einzelnen Strukturen keine Kreisverweise vorkommen (im einfachsten Fall: Element 1 verweist auf Element 2 und dieses wieder auf Element 1). ~~Zu jedem Element führt also nur genau ein Weg~~, die einzelnen Elemente bilden einen Binärbaum.

Bonusaufgabe

Für die Unterstützung von Kreisverweisen (eine Wahl kann den Spieler also zu einem früheren Zeitpunkt im Spiel zurückbringen) erhalten Sie (je nach Qualität Ihrer Umsetzung) bis zu 5 Bonuspunkte. Bonuspunkte sollen, um die vollen 5 Punkte zu bekommen folgendes beinhalten:

- Eine Referenzierung auf sich selbst erkennen.
- Kreisverweise im Baum erkennen.
- Erkennen dass es einen Kreis gibt jedoch auch ein Ende und somit das Spiel normal starten.
- Erkennen wenn ein File mehrmals vorkommt und diese dann zusammenlegen (symlinks, Kopien, andere Pfade, etc.)

Ausgabe

Für jedes einzelne Element lautet die Ausgabe wie folgt:

```
------------------------------\n
[Titel des Elements]\n
\n
[Anzeigetext]\n
\n
Deine Wahl (A/B)?
```

Anmerkungen: Beachten Sie das Leerzeichen zum Abschluss. Die erste Zeile besteht aus 30 '-'.

Im Falle, dass ein Ende der Geschichte erreicht wurde, lautet die Ausgabe wie folgt:

```
------------------------------\n
[Titel des Elements]\n
\n
[Anzeigetext]\n
\n
ENDE\n
```

Das Programm wird in diesem Fall mit dem Rückgabewert 0 beendet.

Eingabe

Im Anschluss an die Ausgabe des Elements muss der Benutzer "A\n" oder "B\n" eingeben. Das entsprechende nächste Element wird ausgegeben (s.o.).

Bei der Eingabe von einem EOF soll das Programm beendet werden.

Fehlermeldungen

Reihenfolge der Fehlermeldungen

Die Priorität der Fehlermeldungen entspricht der Reihenfolge, in der sie genannt werden. Sollten also mehrere Fehlermeldungen auf eine Eingabe zutreffen, soll nur die wichtigste ausgegeben und der Befehl danach abgebrochen werden.

Rückgabewerte und programmbeendende Fehlermeldungen

- Bei falschem Aufrufen des Programms (z.B. zu viele Argumente):
  - ```
    Usage: ./ass2 [file-name]\n
    ```
  - Rückgabewert: 1
- Geht während der Laufzeit der Speicher aus, so soll Folgendes angezeigt und returniert werden:
  - ```
    [ERR] Out of memory.\n
    ```
  - Rückgabewert: 2
- Sollte beim Einlesen der Dateien beim Programmstart ein Fehler auftreten (korruptes oder nicht existierendes File), so muss das Programm wie folgt abgebrochen werden:
  - ```
    [ERR] Could not read file [filename].\n
    ```
  - Rückgabewert: 3

Nicht programmabbrechende Fehlermeldungen

- Sollte der Benutzer bei der Eingabe seiner Wahl etwas anderes als A oder B eingeben.
  - ```
    [ERR] Please enter A or B.\n
    ```

Spezifikation

Einschränkungen

- Alle Dateien müssen bereits beim Programmstart eingelesen werden (also kein sukzessives Einlesen je nach Eingabe des Spielers)
- keine weiteren Ausgaben
- alle Ausgaben müssen auf stdout erfolgen

Erlaubte Bibliotheken

- alle C Standard Bibliotheken

Abgabe

- Dateinamen laut Abgabenliste
- Archiv beinhaltet keine Verzeichnisse oder andere Dateien
- Abgabe bis spätestens 7.12.2017 14:00:00 Uhr

Abgabenliste

- Quellcode (ass2.c) in einem .tar.gz oder .zip Archiv (ass2.tar.gz oder ass2.zip)

Verantwortliche Tutoren

- Florian Bernhardt
- Marcel Nageler