Entwicklung einer Computermorphologie

In dieser Übung sollen Sie eine einfache Morphologie für eine beliebige Sprache (außer Englisch, Chinesisch, Deutsch) implementieren. Die morphologische Analyse eines Wortes soll die Grundform und die morphologischen Merkmale liefern, also für das Wort *glaubtest* bspw. die Analyse `glauben<V><past><ind><2><sg>`. Für die Implementierung sollen die SFST-Werkzeuge verwendet werden:
`http://www.cis.lmu.de/~schmid/tools/SFST`

Schritte:

- Finden Sie heraus, welche **Flexionsklassen** in der Sprache existieren. Hier sind Wörterbücher und Webseiten wie Wiktionary hilfreich.
- Wählen Sie eine Teilmenge der Flexionsklassen aus (insgesamt mindestens 3 Flexionsklassen aus mindestens 2 Wortarten).
- Implementieren Sie die Flexionsklassen und die erforderlichen **morphophonologischen Regeln**. (Mindestens eine selbst geschriebene Regel muss dabei sein.) Zusammengesetzte Flexionsformen wie *"hat gesprochen"* müssen Sie nicht behandeln.

  Empfohlene Schritte:

  - Erstellen Sie eine Datei mit mindestens 5 **Stammlexikon**-Einträgen für jede Flexionsklasse. Sie können hier auch Stämme (glaub) auf Grundformen (glauben) abbilden durch einen Eintrag der Form: `glaube:<>n:<>`

    Beachten Sie dabei, dass in einer Lexikondatei nur die Symbole ":","<",">" und "\" als Operatoren interpretiert werden. Sie können also nicht schreiben `glaub{en}:{}`

    Sie können die Verb-Endungen aber auch so entfernen:
    `$verbstems$ = "verbs.lex" || [a-zäöüß]* {en}:{}`
  - Erstellen Sie für jede Flexionsklasse einen **Flexions-Transducer**, der morphologische Merkmale auf Flexionsendungen abbildet, bspw. `{<pres><1><sg>}:{e} | {<pres><2><sg>}:{st}|...`
  - Lesen Sie jedes Stammlexikon in einen Transducer ein und **konkatenieren** Sie diesen Transducer mit dem entsprechenden Flexions-Transducer.
  - **Vereinigen** Sie die Transducer für die verschiedenen Flexionsklassen mit dem Disjunktions-Operator.
  - Schreiben Sie die notwendigen **morphophonologischen Regeln**, um die korrekten Wortformen zu erhalten (bspw. für happy+er → happier), und wenden Sie sie an.

    Rat: Verwenden sie lieber den Operator `^->` als den Operator `<=>`, da er mächtiger und einfacher anzuwenden ist.
- Kommentieren Sie Ihren Code. Schreiben Sie zu jeder Ersetzungsregel ein Beispiel für die gewünschte Ersetzung als Kommentar dazu, bspw. `try<V>s --> trie<V>s`

- Dokumentieren Sie, welche Phänomene Sie erfasst haben, und an welcher Quelle (Buch, Webseite) Sie sich orientiert haben.

Nennen Sie Ihr SFST-Programm <language>-morph.fst (wobei <language> der Name Ihrer gewählten Sprache ist) und schicken Sie es mit Ihrer Dokumentation in einer zip-Datei an `schmid@cis.lmu.de`.

Tipp

Bei morphophonologischen Regeln sind meistens Triggersymbole nützlich. Sie erlauben es, die Anwendung der Regeln genau zu kontrollieren, so dass sie nicht an falscher Stelle angewendet werden können.

Beispiel:

Deutsche Nomen wie "Haus" oder "Buch" bilden den Plural mit Umlautung. Hier kann man zu der Endung -er ein Triggersymbol `<UL>` hinzufügen: `{<pl>}:{<UL>er}` und dann später (nach der Konkatenation von Stämmen und Endungen) eine morphophonologische Regel anwenden, die den Stamm nur dann umlautet, wenn das Triggersymbol nachfolgt. Gleichzeitig oder anschließend wird das Triggersymbol gelöscht. Das folgende Beispiel benutzt temporär das Symbol `<B>`, um den Wortanfang zu markieren und löscht es am Ende wieder.

```
ALPHABET = [A-Za-zäöüÄÖÜß] <UL>:<> <B>:<>
$cons$ = [bcdfghjklmnpqrstvwxz]
$Umlaut$ = ([AOUaou]:[ÄÖÜäöü]|{aa}:ä|{oo}:ö|{au}:{äu}) ^-> \
           (($cons$|<B>) __ $cons$* (e[rl])? <UL>)
$Morph$ = <>:<B> $Morph$ || $Umlaut$
```