Exercise: Southern Brazilian Portuguese

The task is to create a rule system that maps from orthographical strings in Portuguese (this will be the lexical level) to strings that represent their pronunciation (this will be the surface side).
A sample mapping of written "caso" to spoken "kazu" looks like this, with, by convention, the lexical string on top and the surface string on the bottom.

Lexical:	caso
Surface:	kazu

After studying the facts listed below, write your grammar as an xfst script (named something like portuguese.script) compile it and test it using the apply down command in xfst.

Standard Portuguese orthography is not always a complete guide to the pronunciation of a word (especially in the case of the letter "x" and the vowels written "o" and “e”), so we will restrict and simplify the data slightly.

The Facts to be Modeled

- The following description is based on the rather conservative pronunciation of Portuguese in Porto Alegre, Rio Grande do Sul, Brazil. Because the orthography is even more conservative, the rules will roughly characterize the phonological changes that have occurred in one dialect since the orthography fossilized.
- Your grammar should generate from lexical strings (using apply down) like the following, written in standard Brazilian Portuguese orthography. We will limit the input to lowercase words in this exercise.

  casa
  cimento
  me
  disse
  peruca
  simpático
  braço
  árvore

- The surface level produced by your grammar will be a kind of crude phonemic alphabet, with the following extra symbols:

J	palatalized "d", similar to the phoneme spelled "j" in English "judge".
C	palatalized "t", similar to the phoneme spelled "ch" in "church".
$	alveopalatal sibilant, like the phoneme spelled "sh" in English "ship".
L	phoneme spelled "lh" in Portuguese "filho" (or "gli" in Italian "figlio").
N	phoneme spelled "nh" in Portuguese "ninho" (like the French "gn" in "digne").
R	phoneme written "rr" inside words, single "r" at the beginning.

Because we have limited our input words to lowercase letters, the six special characters will appear only in surface strings, never at the lexical level. The dollar sign $ character is special in regular expressions, so precede it with a percent sign (%) to literalize it or put it in double quotes.

- The mapping from orthography (lexical side) to pronunciation (surface side) includes the following:

  - The orthographical (lexical-side) "ç" is always pronounced /s/[1]; in other words, a "ç" on top always corresponds to an "s" on the bottom.

    braço
    brasu
  - The orthographical "ss" is always pronounced /s/. In this and following illustrations, the lexical and surface strings are lined up character pair by character pair, with the 0 (zero, also called epsilon) filling out the places where a lexical symbol maps to the empty string. These zeros will not appear in the final output.

    interesse
    interes0i
  - The orthographical "c" before "e" or "i" (or accented versions of these vowel letters) is always pronounced /s/.

    cimento
    simentu
  - The orthographical digraph "ch" is pronounced /$/.

    chato
    $0atu
  - Elsewhere (i.e. not ch), orthographical "c" is always pronounced /k/.

    casa
    kasa
  - No "c" should appear in surface strings.
  - The orthographical digraph "lh" is realized as /L/.

    filho
    fiL0u
  - The orthographical digraph "nh" is realized as /N/.

    ninho
    niN0u
  - Elsewhere, "h" is silent and is simply realized as 0 (zero, the empty string).

    homem
    0omem
  - The orthographical digraph "rr" is always realized as /R/. Also, the single r at the beginning of a word is always realized as /R/. Elsewhere, r:r, i.e. lexical "r" is realized as /r/.

carro	rápido	caro	cantar
kaR0u	Rapidu	karu	kantar
  - The unaccented "e" is pronounced /i/ at the end of a word, and when it appears in the context between "p" and "r" at the beginning of a word; e.g.

peruca	case
piruka	kazi
  - Between the lexical "e" and the end of word there can be an optional "s". Elsewhere e:e.

    cases
    kazis
  - An "o" is pronounced /u/ at the end of a word.

    braço
    brasu
  - Between the lexical "o" and the end of word there can be an optional "s". Elsewhere o:o.

[1] In these explanations, we follow the IPA convention of indicating phonemes (lower-side symbols) between slashes. These slashes will not really appear in the output string.

braços
brasus

- A single "s" is pronounced /z/ when it appears between two vowels.

camisa	case
kamiza	kazi

- Elsewhere s:s (but see above where s s -> s).
- A word-final "z" is realized as /s/.

vez
ves

- Elsewhere, z:z.
- A "d" is pronounced /J/ when it appears before a SURFACE sound /i/. (N.B. This change occurs in the environment of any SURFACE /i/, no matter what that surface /i/ may have been at the lexical level.) Elsewhere d:d.

lisse	verdade	paredes
Jis0i	verdaJi	pareJis

- A "t" is pronounced /C/ when it appears before a surface sound /i/. (N.B. This change occurs in the environment of any SURFACE /i/, no matter what that surface /i/ may have been at the lexical level.) Elsewhere t:t.

tio	partes
Ciu	parCis

- The vowels are a, e, i, o, u, á, é, í, ó, ú, ã, õ, â, ê, ô, ü and à. All lexical symbols map to themselves on the surface level by default.

Testing Portuguese Pronunciation

Write a set of that performs the mappings indicated. As in the kaNpat example, the rules should be organized in a cascade, with the composition operator (.o.) between the rules. Be very careful about ordering your rules correctly; the rules cannot be expressed in exactly the same order as the facts listed just above. Compile the rules using the read regex from utility in and test them using the apply down utility.

You should be able to handle the following examples, entering the lexical (top) string in each case and getting back the surface (bottom) string. (The zeros are not shown here and should not appear in your output.) To facilitate the testing, you can type all the input (upper-side) words into a file, called something like mydata, and tell apply down to read the various input strings from that file.

```
xfst[1] apply down < mydata
```

disse	peru	pedaço	livro	parte	parede	sabe	cada
Jisi	piru	pedasu	livru	parCi	pareJi	sabi	kada

simpático	verdade	casa	braço	chato	vermelho	gatinho	filhos
simpáCiku	verdaJi	kaza	brasu	$atu	vermeLu	gaCiNu	fiLus

luz	case	braços	partes	paredes	me	antes	ninhos
lus	kazi	brasus	parCis	pareJis	mi	anCis	niNus

Be sure to test ALL the examples to make sure that your rules are really working as they should. Modify your rules and re-apply the input words until the grammar is working perfectly.