(1982/2017). In A. Bunger (Ed.)., *Special volume: Reissue of Innovations in Linguistic Education Volume 2. IULC Working Papers in Linguistics*, Vol. 17, No. 3.

TEACHING BASIC PHONOLOGY THROUGH PHONOLOGY IN BASIC

Donald A. Becker

Professor of German & Linguistics
University of Wisconsin-Madison

0. The contributors to the glossy computer hobbyist publications that dot the magazine racks of bookstores and supermarkets these days in ever increasing numbers would have us believe that the day is rapidly approaching when personal computers will be as common in American households as toasters. While such extravagant claims are admittedly somewhat suspect, the booming sales of TRS-80's, Apple II's, Commodore Pets, and other home computers should make educators in all fields begin to devote some serious thought to how they might modify their teaching in order to reap the greatest pedagogical rewards from the unprecedented availability of sophisticated computer hardware and the resultant growth in their average student's computer literacy. Linguists especially should, I think, be quick to realize that the machines they buy to balance their checkbooks, teach their children arithmetic, and serve as their chess partners can prove quite useful in their teaching as well.

One area of linguistics in which personal computers can be put to good use is phonology. This is hardly surprising in view of the remarkable resemblance which phonological descriptions (at least those cast in the generative mold) bear to computer programs. Until recently, however, only those phonologists who were simultaneously specialists in computer science had access to machines that could process their 'software' while the vast majority of their colleagues were forced to 'run' derivations by hand.

To illustrate how a phonologist might use a personal computer in his or her work, I shall present, in Section 1 of this article, a set of seventeen phonological rules formulated in distinctive feature notation and containing most of the abbreviative devices commonly found in the generative phonological literature. In Section 2 these rules will be translated into Radio Shack's TRS-80 Level II BASIC and incorporated into a program that will run derivations in which the rules are applied in a specified order. Section 3 will provide some sample derivations which can be generated with this program. Section 4 will supply some technical notes about the program. Finally, in Section 5, I shall outline some of the pedagogical applications for which such a program is suited.

1. The Rules

It will be assumed that the phonological rules presented in this section apply to underlying representations composed of the segments and boundaries defined in classificatory matrix (19). The distinctive features abbreviated in that matrix are to be interpreted as follows: SE (segment), WB (word boundary), CS (consonantal), OB (obstruent), SY (syllabic), CT (continuant), AN (anterior), CO (coronal), VO (voiced), HI (high), LO (low), BA (back), RN (round), TN (tense), NA (nasal), ST (stress). The symbols chosen to represent the segments in (19) are all ones that can be typed on the keyboard of a TRS-80 home computer equipped with Radio Shack's lower case modification kit. Those symbols which do not belong to the International Phonetic Alphabet are explained in (18). It will be noted that I am using the signs + and # to represent a word-internal morpheme boundary and a word boundary, respectively. Phrase boundaries will be marked by a cluster of two or more #'s. Unless otherwise indicated by the label 'morpheme-level' or 'phrase-level', each rule will be regarded as applying within the domain of the word.

(1) Obstruent Dissimilation

$$[+\text{obstruent}] \rightarrow [-\alpha\text{continuant}] / \_\_ \begin{bmatrix} +\text{obstruent} \\ \alpha\text{continuant} \end{bmatrix}$$

I.e., if the second of two contiguous obstruents is a stop, the first must be a fricative, but if the second is a fricative, the first must be a stop.

(2) High Back Vowel Lowering

$$\begin{bmatrix} +\text{syllabic} \\ -\text{low} \\ +\text{back} \end{bmatrix} \rightarrow [-\text{high}] / \_\_ \left(\begin{bmatrix} -\text{syllabic} \\ -\text{nasal} \end{bmatrix} \right) \begin{bmatrix} +\text{syllabic} \\ +\text{low} \\ +\text{back} \end{bmatrix}$$

I.e., [u] is lowered to [o] and [ɨ] is lowered to [ʌ] before an [a] that is either contiguous to the high vowel or separated from it by a single nonnasal consonant.

(3) Front Vowel Rounding

$$\begin{bmatrix} +\text{syllabic} \\ -\text{low} \\ -\text{back} \end{bmatrix} \rightarrow [+\text{round}] / \begin{bmatrix} +\text{anterior} \\ -\text{coronal} \end{bmatrix} \_\_$$

I.e., a high or mid front vowel is rounded if it stands immediately after a labial consonant.

(4) Consonant Voicing

$$[+\text{consonantal}] \rightarrow [+\text{voice}] / [+\text{voice}] \_\_ [+\text{voice}]$$

I.e., a voiceless consonant other than [h] or [ʔ] is voiced in voiced surroundings.

(5) Stress Placement

$$[+\text{syllabic}] \rightarrow [+\text{stress}] / \_\_ C_0 (CV)_0 \#$$

I.e., the last closed syllable of a word is stressed, or, if there are no closed syllables, the first syllable is stressed.

(6) h-Deletion

$$\begin{bmatrix} -\text{consonantal} \\ +\text{continuant} \\ -\text{voice} \end{bmatrix} \rightarrow \emptyset / [-\text{syllabic}] \_\_$$

I.e., [h] is deleted after a contiguous consonant.

(7) Word-final Obstruent Devoicing

[+obstruent] → [-voice]/__#

I.e., an obstruent is devoiced in word-final position.

(8) Geminate Vowel Simplification

$$\begin{bmatrix} +\text{syllabic} \\ \alpha\text{high} \\ \beta\text{low} \\ \gamma\text{back} \\ \delta\text{round} \end{bmatrix} \rightarrow \emptyset / \begin{bmatrix} +\text{syllabic} \\ \alpha\text{high} \\ \beta\text{low} \\ \gamma\text{back} \\ \delta\text{round} \end{bmatrix} \_\_$$

I.e., a vowel is deleted if it stands immediately after a vowel that shares its specifications for the features high, low, back, and round.

(9) Metathesis

$$\begin{bmatrix} +\text{consonantal} \\ -\text{obstruent} \\ -\text{nasal} \end{bmatrix}_1 [+\text{obstruent}]_2 \rightarrow 2\ 1$$

I.e., a liquid and a following obstruent are metathesized.

(10) Nasal Assimilation

$$\begin{bmatrix} -\text{syllabic} \\ +\text{nasal} \end{bmatrix} \rightarrow \begin{bmatrix} \alpha\text{anterior} \\ \beta\text{coronal} \\ \gamma\text{high} \\ \delta\text{back} \end{bmatrix} / \_\_ \begin{bmatrix} +\text{consonantal} \\ +\text{obstruent} \\ -\text{continuant} \\ \alpha\text{anterior} \\ \beta\text{coronal} \\ \gamma\text{high} \\ \delta\text{back} \end{bmatrix}$$

I.e., a nasal consonant assumes the point of articulation of an immediately following oral stop other than [ʔ].

(11) i-Umlaut

$$[+\text{syllabic}] \rightarrow [-\text{back}] / \_\_ C_0^2 \begin{bmatrix} +\text{syllabic} \\ +\text{high} \\ -\text{back} \end{bmatrix}$$

I.e., a back vowel is fronted before a syllable containing a high front vowel if no more than two consonants intervene.

(12) Alpha-Switching

$$\begin{bmatrix} +\text{syllabic} \\ +\text{back} \\ +\text{round} \\ \alpha\text{high} \end{bmatrix} \rightarrow \begin{bmatrix} +\text{high} \\ -\alpha\text{back} \end{bmatrix}$$

I.e., [u] is fronted to [y] and [o] is raised to [u].

(13) Nasalization

$$\text{(Morpheme-Level)} \quad \underset{1}{[+\text{syllabic}]}\underset{2}{[+\text{nasal}]}\underset{3}{C} \rightarrow \underset{1}{[+\text{nasal}]}\underset{2}{\emptyset}\ 3$$

I.e., a nasal consonant which stands between a preceding vowel and a following consonant is deleted with compensatory nasalization of the vowel, provided that all three segments belong to the same morpheme.

(14) Vowel Insertion

$$\text{(Phrase-Level)} \quad \emptyset \rightarrow a / [+\text{obstruent}]\_\_[+\text{obstruent}]$$

I.e., within a phrase, an [a] is inserted between two obstruents even if the latter belong to different words.

(15) Nasalized Vowel Laxing

$$\begin{bmatrix} +\text{syllabic} \\ +\text{nasal} \\ -\text{back} \\ -\text{low} \\ \langle -\text{stress} \rangle \end{bmatrix} \rightarrow \begin{bmatrix} -\text{tense} \\ \langle -\text{high} \rangle \end{bmatrix}$$

I.e., a nasalized high or mid front vowel is laxed and, if unstressed and high, lowered to mid.

(16) Palatalization

$$\begin{bmatrix} +\text{consonantal} \\ +\text{obstruent} \\ +\text{continuant} \\ \begin{Bmatrix} -\text{anterior} \\ +\text{coronal} \end{Bmatrix} \end{bmatrix} \rightarrow \begin{bmatrix} -\text{anterior} \\ +\text{coronal} \\ +\text{high} \\ -\text{back} \end{bmatrix} / \_\_ \begin{bmatrix} +\text{syllabic} \\ +\text{high} \\ -\text{back} \end{bmatrix}$$

I.e., [s] and [x] become [SH] and their voiced counterparts [z] and [GH] become [ZH] before a high front vowel.

(17) Glottal Stop Insertion

(Phrase-Level) $\emptyset \rightarrow$?/[+syllabic]#__[+syllabic]

I.e., a glottal stop is inserted before a word-initial vowel if an immediately preceding word within the same phrase ends in a vowel.

(18) Definitions of unusual phonetic symbols

Symbols	Definitions
NG	a voiced velar nasal
@	a low front unrounded vowel
ɨ	a mid central unrounded vowel
ø	a tense mid front rounded vowel
GH	a voiced velar fricative
SH	a voiceless palatoalveolar fricative
ZH	a voiced palatoalveolar fricative
?	a voiceless glottal stop
!	a high central unrounded vowel
I	a lax high front unrounded vowel
Y	a lax high front rounded vowel
E	a lax mid front unrounded vowel
OE	a lax mid front rounded vowel
M	a diacritic indicating that the preceding vowel is nasalized
'	a diacritic indicating that the preceding vowel is stressed

(19) Classificatory matrix

	#	+	a	i	u	p	t	k	b	d	g	f	s	x	e	o	m	n	NG	@	↑	l	r	j	w	y	ø	v	z	GH	SH	ZH
SE	−	−	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
WB	+	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
CS			−	−	−	+	+	+	+	+	+	+	+	−	−	+	+	+	−	−	−	+	+	−	−	−	−	+	+	+	+	+
OB			−	−	−	+	+	+	+	+	+	+	+	−	−	−	−	−	−	−	−	−	−	−	−	−	−	+	+	+	+	+
SY			+	+	+	−	−	−	−	−	−	−	−	−	+	+	−	−	−	+	+	−	−	−	−	+	+	−	−	−	−	−
CT			+	+	+	−	−	−	−	−	−	+	+	+	+	+	−	−	−	+	+	+	−	+	+	+	+	+	+	+	+	+
AN			−	−	−	+	+	−	+	+	−	+	+	−	−	−	+	+	−	−	−	+	+	−	−	−	−	+	+	−	−	−
CO			−	−	−	−	+	−	−	+	−	−	+	−	−	−	−	+	−	−	−	+	+	−	−	−	−	+	−	+	+	+
VO			+	+	+	−	−	−	+	+	+	−	−	−	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	−	+	+
HI			−	+	+	−	−	+	−	−	+	−	−	+	−	−	−	−	+	−	−	−	−	+	+	+	−	−	−	+	+	+
LO			+	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	+	−	−	−	−	−	−	−	−	−	−	−	−
BA			+	−	+	−	−	+	−	−	+	−	+	−	−	+	−	+	−	−	−	+	−	−	−	+	−	−	−	+	−	−
RN			−	−	+	−	−	−	−	−	−	−	−	+	−	−	−	−	−	−	−	−	−	+	+	+	−	−	−	−	−	−
TN			+	+	+	−	−	−	−	−	−	−	−	+	+	−	−	−	−	+	+	−	−	+	+	+	−	−	−	−	−	−
NA			−	−	−	−	−	−	−	−	−	−	−	−	−	+	+	+	−	−	−	−	−	−	−	−	−	−	−	−	−	−
ST			−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−

	?	h	aM	ĭM	uM	eM	oM	@M	yM	øM	↑M	!	!M	a′	aM′	i′	ĭM′	u′	uM′	e′	eM′	o′
SE	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
WB	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
CS	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
OB	+	+	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
SY	−	−	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
CT	−	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
AN	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
CO	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
VO	−	−	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
HI	−	−	−	+	+	−	−	−	+	−	−	+	+	−	−	+	+	+	+	−	−	−
LO	+	+	+	−	−	−	−	+	−	−	−	−	−	+	+	−	−	−	−	−	−	−
BA	−	−	+	−	+	−	+	−	−	−	+	+	+	+	+	−	−	+	+	−	−	+
RN	−	−	−	−	+	−	+	−	+	+	−	−	−	−	−	−	−	+	+	−	−	+
TN	−	−	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
NA	−	−	+	+	+	+	+	+	+	+	+	−	+	−	+	−	+	−	+	−	+	−
ST	−	−	−	−	−	−	−	−	−	−	−	−	−	+	+	+	+	+	+	+	+	+

	oM′	y′	yM′	ø′	øM′	↑′	↑M′	@′	@M′	!′	!M′	IM	YM	EM	OEM	IM′	YM′	EM′	OEM′
SE	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
WB	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
CS	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
OB	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
SY	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
CT	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
AN	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
CO	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−	−
VO	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
HI	−	+	+	−	−	−	−	−	−	+	+	+	+	−	−	+	+	−	−
LO	−	−	−	−	−	−	−	+	+	−	−	−	−	−	−	−	−	−	−
BA	+	−	−	−	−	+	+	−	−	+	+	−	−	−	−	−	−	−	−
RN	+	+	+	+	+	−	−	−	−	−	−	−	+	−	+	−	+	−	+
TN	+	+	+	+	+	+	+	+	+	+	+	−	−	−	−	−	−	−	−
NA	+	−	+	−	+	−	+	−	+	−	+	+	+	+	+	+	+	+	+
ST	+	+	+	+	+	+	+	+	+	+	+	−	−	−	−	+	+	+	+

2. Program Listing

```
 1Ø REM *Generative Phonological Derivations*
 2Ø CLEAR 2ØØ:CLS
 3Ø DEFINT A-Z
 4Ø DIM SG$(3Ø),SE(3Ø),WB(3Ø),CS(3Ø),OB(3Ø),SY(3Ø)
 5Ø DIM CT(3Ø),AN(3Ø),CO(3Ø),VO(3Ø),HI(3Ø),LO(3Ø)
 6Ø DIM BA(3Ø),RN(3Ø),TN(3Ø),NA(3Ø),ST(3Ø)
1ØØ DATA #,-1,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
1Ø1 DATA +,-1,-1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
1Ø2 DATA a,1,-1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,1,-1,-1
1Ø3 DATA i,1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,-1,1,-1,-1
1Ø4 DATA u,1,-1,-1,-1,1,1,-1,-1,1,1,-1,1,1,1,-1,-1
1Ø5 DATA p,1,-1,1,1,-1,-1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1
1Ø6 DATA t,1,-1,1,1,-1,-1,1,1,-1,-1,-1,-1,-1,-1,-1,-1
1Ø7 DATA k,1,-1,1,1,-1,-1,-1,-1,-1,1,-1,1,-1,-1,-1,-1
1Ø8 DATA b,1,-1,1,1,-1,-1,1,-1,1,-1,-1,-1,-1,-1,-1,-1
1Ø9 DATA d,1,-1,1,1,-1,-1,1,1,1,-1,-1,-1,-1,-1,-1,-1
11Ø DATA g,1,-1,1,1,-1,-1,-1,-1,1,1,-1,1,-1,-1,-1,-1
111 DATA f,1,-1,1,1,-1,1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1
112 DATA s,1,-1,1,1,-1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1
113 DATA x,1,-1,1,1,-1,1,-1,-1,-1,1,-1,1,-1,-1,-1,-1
114 DATA e,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,-1,1,-1,-1
115 DATA o,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,1,1,1,-1,-1
116 DATA m,1,-1,1,-1,-1,-1,1,-1,1,-1,-1,-1,-1,-1,1,-1
117 DATA n,1,-1,1,-1,-1,-1,1,1,1,-1,-1,-1,-1,-1,1,-1
118 DATA NG,1,-1,1,-1,-1,-1,-1,-1,1,1,-1,1,-1,-1,1,-1
119 DATA @,1,-1,-1,-1,1,1,-1,-1,1,-1,1,-1,-1,1,-1,-1
12Ø DATA ↑,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,1,-1,1,-1,-1
121 DATA l,1,-1,1,-1,-1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1
122 DATA r,1,-1,1,-1,-1,-1,1,1,1,-1,-1,-1,-1,-1,-1,-1
123 DATA j,1,-1,-1,-1,-1,1,-1,-1,1,1,-1,-1,-1,1,-1,-1
124 DATA w,1,-1,-1,-1,-1,1,-1,-1,1,1,-1,1,1,1,-1,-1
125 DATA y,1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1
126 DATA Ø,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,1,1,-1,-1
127 DATA v,1,-1,1,1,-1,1,1,-1,1,-1,-1,-1,-1,-1,-1,-1
128 DATA z,1,-1,1,1,-1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1
129 DATA GH,1,-1,1,1,-1,1,-1,-1,1,1,-1,1,-1,-1,-1,-1
13Ø DATA SH,1,-1,1,1,-1,1,-1,1,-1,1,-1,-1,-1,-1,-1,-1
131 DATA ZH,1,-1,1,1,-1,1,-1,1,1,1,-1,-1,-1,-1,-1,-1
132 DATA ?,1,-1,-1,1,-1,-1,-1,-1,-1,-1,1,-1,-1,-1,-1,-1
133 DATA h,1,-1,-1,1,-1,1,-1,-1,-1,-1,1,-1,-1,-1,-1,-1
134 DATA aM,1,-1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,1,1,-1
135 DATA iM,1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,-1,1,1,-1
136 DATA uM,1,-1,-1,-1,1,1,-1,-1,1,1,-1,1,1,1,1,-1
137 DATA eM,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,-1,1,1,-1
138 DATA oM,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,1,1,1,1,-1
139 DATA @M,1,-1,-1,-1,1,1,-1,-1,1,-1,1,-1,-1,1,1,-1
14Ø DATA yM,1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,1,-1
141 DATA ØM,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,1,1,1,-1
142 DATA ↑M,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,1,-1,1,1,-1
```

```
143 DATA !,1,-1,-1,-1,1,1,-1,-1,1,1,-1,1,-1,1,-1,-1
144 DATA !M,1,-1,-1,-1,1,1,-1,-1,1,1,-1,1,-1,1,1,-1
145 DATA a',1,-1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,1,-1,1
146 DATA aM',1,-1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,1,1,1
147 DATA i',1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,-1,1,-1,1
148 DATA iM',1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,-1,1,1,1
149 DATA u',1,-1,-1,-1,1,1,-1,-1,1,1,-1,1,1,1,-1,1
15Ø DATA uM',1,-1,-1,-1,1,1,-1,-1,1,1,-1,1,1,1,1,1
151 DATA e',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,-1,1,-1,1
152 DATA eM',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,-1,1,1,1
153 DATA o',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,1,1,1,-1,1
154 DATA oM',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,1,1,1,1,1
155 DATA y',1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,1
156 DATA yM',1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,1,1
157 DATA Ø',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,1,1,-1,1
158 DATA ØM',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,1,1,1,1
159 DATA †',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,1,-1,1,-1,1
16Ø DATA †M',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,1,-1,1,1,1
161 DATA @',1,-1,-1,-1,1,1,-1,-1,1,-1,1,-1,-1,1,-1,1
162 DATA @M',1,-1,-1,-1,1,1,-1,-1,1,-1,1,-1,-1,1,1,1
163 DATA !',1,-1,-1,-1,1,1,-1,-1,1,1,-1,1,-1,1,-1,1
164 DATA !M',1,-1,-1,-1,1,1,-1,-1,1,1,-1,1,-1,1,1,1
165 DATA IM,1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,-1,-1,1,-1
166 DATA YM,1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,-1,1,-1
167 DATA EM,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,-1,-1,1,-1
168 DATA OEM,1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,1,-1,1,-1
169 DATA IM',1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,-1,-1,1,1
17Ø DATA YM',1,-1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,-1,1,1
171 DATA EM',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,-1,-1,1,1
172 DATA OEM',1,-1,-1,-1,1,1,-1,-1,1,-1,-1,-1,1,-1,1,1
2ØØ T$="Underlying Representation"
21Ø PRINT "Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)"
22Ø PRINT "to separate each segment & boundary from its neighbors"
23Ø INPUT UR$
24Ø W$="#/#/"+UR$+"/#/#/"
25Ø T=Ø:A=1:G=26
26Ø FOR B=1 TO 4
27Ø IF A+B-1>LEN(W$) GOTO 38Ø
28Ø B$(B)=MID$(W$,A+B-1,1)
29Ø IF B$(B)="/" GOTO 32Ø
3ØØ C$=C$+B$(B)
31Ø NEXT B
32Ø T=T+1
33Ø SG$(T)=C$
34Ø A=A+B
35Ø C$=""
37Ø GOTO 26Ø
38Ø GOSUB 12ØØØ
39Ø PRINT STRING$(G-LEN(T$)," ")+T$;": ";Z$
4ØØ W$=Z$
7ØØ FOR H=1 TO 17
71Ø ON H GOTO 25ØØ,21ØØ,1ØØØ,11ØØ,12ØØ,13ØØ,14ØØ,15ØØ,16ØØ,17ØØ,18ØØ,19ØØ,
    2ØØØ,22ØØ,23ØØ,24ØØ,26ØØ
```

```
 72Ø PRINT STRING$(G-LEN(T$)," ")+T$;": ";
 73Ø IF Z$ <> W$ PRINT Z$ ELSE PRINT STRING$(LEN(Z$),"-")
 74Ø W$=Z$
 75Ø NEXT H
 76Ø GOTO 999Ø
 99Ø REM *Phonological Rules*
1ØØØ T$="Obstruent Dissimilation"
1Ø1Ø FOR K=1 TO T
1Ø15 IF SG$(K+1)="+" THEN L=2 ELSE L=1
1Ø2Ø IF OB(K)=1 AND OB(K+L)=1 GOSUB 1Ø5Ø
1Ø3Ø NEXT K
1Ø4Ø GOTO 72Ø
1Ø5Ø CT(K)=CT(K+L)*-1
1Ø6Ø GOSUB 15ØØØ
1Ø7Ø RETURN
11ØØ T$="High Back Vowel Lowering"
111Ø FOR K=1 TO T
1112 M=Ø
1115 FOR L=1 TO 2
1117 IF SG$(K+L+M)="+" THEN M=M+1
1118 IF SY(K+L+M)=-1 AND NA(K+L+M)=1 GOTO 114Ø
112Ø IF SY(K)=1 AND SY(K+L+M)=1 GOTO 115Ø
113Ø NEXT L
114Ø NEXT K
1145 GOTO 72Ø
115Ø IF LO(K)=-1 AND BA(K)=1 AND LO(K+L+M)=1 AND BA(K+L+M)=1 GOTO 117Ø
116Ø GOTO 114Ø
117Ø HI(K)=-1
118Ø GOSUB 15ØØØ
119Ø GOTO 114Ø
12ØØ T$="Front Vowel Rounding"
121Ø FOR K=1 TO T
1215 IF SG$(K-1)="+" THEN L=2 ELSE L=1
122Ø IF SY(K)=1 AND LO(K)=-1 AND BA(K)=-1 AND AN(K-L)=1 AND CO(K-L)=-1
     GOSUB 125Ø
123Ø NEXT K
124Ø GOTO 72Ø
125Ø RN(K)=1
126Ø GOSUB 15ØØØ
127Ø RETURN
13ØØ T$="Consonant Voicing"
131Ø FOR K=1 TO T
1315 IF SG$(K-1)="+" THEN L=2 ELSE L=1
1316 IF SG$(K+1)="+" THEN M=2 ELSE M=1
132Ø IF VO(K-L)=1 AND VO(K+M)=1 AND CS(K)=1 GOSUB 135Ø
133Ø NEXT K
134Ø GOTO 72Ø
135Ø VO(K)=1
136Ø GOSUB 15ØØØ
137Ø RETURN
14ØØ T$="Stress Placement"
14Ø5 C=Ø:FOR V=1 TO T
14Ø6 IF SG$(V)="#" GOSUB 14Ø9
```

```
14Ø7 NEXT V
14Ø8 GOTO 1412
14Ø9 C=C+1
141Ø C(C)=V
1411 RETURN
1412 FOR PC=1 TO C-1
1413 FOR K=C(PC+1) TO C(PC) STEP -1
1414 IF SG$(K+1)="+" THEN L=2 ELSE L=1
1415 IF SG$(K+L+1)="+" THEN M=2 ELSE M=1
142Ø IF SY(K)=1 AND SY(K+L)=-1 AND SY(K+L+M) <> 1 GOTO 147Ø
143Ø NEXT K
144Ø FOR K=C(PC) TO C(PC+1)
145Ø IF SY(K)=1 GOTO 147Ø
146Ø NEXT K
1465 GOTO 149Ø
147Ø ST(K)=1
148Ø GOSUB 15ØØØ
149Ø NEXT PC
1495 GOTO 72Ø
15ØØ T$="h-Deletion"
151Ø FOR K=1 TO T
1515 IF SG$(K+1)="+" THEN L=2 ELSE L=1
152Ø IF SY(K)=-1 AND CS(K+L)=-1 AND VO(K+L)=-1 AND CT(K+L)=1 GOTO 155Ø
153Ø NEXT K
154Ø GOTO 72Ø
155Ø T=T-1
1551 FOR X=K+L TO T
1552 SG$(X)=SG$(X+1)
1553 NEXT X
156Ø GOSUB 11ØØØ
157Ø GOTO 151Ø
16ØØ T$="Word-final Obst. Devoicing"
161Ø FOR K=1 TO T
162Ø IF OB(K)=1 AND WB(K+1)=1 GOSUB 165Ø
163Ø NEXT K
164Ø GOTO 72Ø
165Ø VO(K)=-1
166Ø GOSUB 15ØØØ
167Ø RETURN
17ØØ T$="Geminate Simplification"
171Ø FOR K=1 TO T
1715 IF SG$(K+1)="+" THEN L=2 ELSE L=1
172Ø IF SY(K)=1 AND SY(K+L)=1 AND HI(K)=HI(K+L) AND LO(K)=LO(K+L) AND
     BA(K)=BA(K+L) AND RN(K)=RN(K+L) GOTO 175Ø
173Ø NEXT K
174Ø GOTO 72Ø
175Ø IF ST(K+L)=1 THEN SG$(K)=SG$(K+L)
1751 T=T-1
1752 FOR X=K+L TO T
1753 SG$(X)=SG$(X+1)
1754 NEXT X
176Ø GOSUB 11ØØØ
```

```
177Ø GOTO 171Ø
18ØØ T$="Metathesis"
181Ø FOR K=1 TO T
1815 IF SG$(K+1)="+" THEN L=2 ELSE L=1
182Ø IF CS(K)=1 AND OB(K)=-1 AND NA(K)=-1 AND OB(K+L)=1 GOTO 185Ø
183Ø NEXT K
184Ø GOTO 72Ø
185Ø M$=SG$(K)
1851 SG$(K)=SG$(K+L)
1852 SG$(K+L)=M$
186Ø GOSUB 12ØØØ
187Ø GOTO 181Ø
19ØØ T$="Nasal Assimilation"
191Ø FOR K=1 TO T
1915 IF SG$(K+1)="+" THEN L=2 ELSE L=1
192Ø IF SY(K)=-1 AND NA(K)=1 AND CS(K+L)=1 AND OB(K+L)=1 AND CT(K+L)
     =-1 GOSUB 195Ø
193Ø NEXT K
194Ø GOTO 72Ø
195Ø AN(K)=AN(K+L)
1951 CO(K)=CO(K+L)
1952 HI(K)=HI(K+L)
1953 BA(K)=BA(K+L)
196Ø GOSUB 15ØØØ
197Ø RETURN
2ØØØ T$="i-Umlaut"
2Ø1Ø FOR K=1 TO T
2Ø12 M=Ø
2Ø15 FOR L=1 TO 3
2Ø17 IF SG$(K+L+M)="+" THEN M=M+1
2Ø18 IF SG$(K+L+M)="#" GOTO 2Ø4Ø
2Ø2Ø IF SY(K)=1 AND SY(K+L+M)=1 GOTO 2Ø5Ø
2Ø3Ø NEXT L
2Ø4Ø NEXT K
2Ø45 GOTO 72Ø
2Ø5Ø IF HI(K+L+M)=1 AND BA(K+L+M)=-1 GOTO 2Ø7Ø
2Ø6Ø GOTO 2Ø4Ø
2Ø7Ø BA(K)=-1
2Ø8Ø GOSUB 15ØØØ
2Ø9Ø GOTO 2Ø4Ø
21ØØ T$="Alpha-Switching"
211Ø FOR K=1 TO T
212Ø IF SY(K)=1 AND BA(K)=1 AND RN(K)=1 GOSUB 215Ø
213Ø NEXT K
214Ø GOTO 72Ø
215Ø BA(K)=HI(K)*-1
2151 HI(K)=1
216Ø GOSUB 15ØØØ
217Ø RETURN
22ØØ T$="Nasalization"
221Ø FOR K=1 TO T
222Ø IF SY(K)=1 AND NA(K+1)=1 AND SY(K+2)=-1 GOTO 225Ø
223Ø NEXT K
```

```
224Ø GOTO 72Ø
225Ø NA(K)=1
2251 GOSUB 15ØØØ
2252 T=T-1
2253 FOR X=K+1 TO T
2254 SG$(X)=SG$(X+1)
2255 NEXT X
226Ø GOSUB 11ØØØ
227Ø GOTO 221Ø
23ØØ T$="Vowel Insertion"
231Ø FOR K=1 TO T
2315 IF SE(K+1)=-1 THEN L=2 ELSE L=1
232Ø IF OB(K)=1 AND OB(K+L)=1 GOTO 235Ø
233Ø NEXT K
234Ø GOTO 72Ø
235Ø FOR X=T TO K+1 STEP -1
2351 SG$(X+1)=SG$(X)
2352 NEXT X
2353 SG$(K+1)="a"
2354 T=T+1
236Ø GOSUB 12ØØØ
237Ø GOTO 231Ø
24ØØ T$="Nasalized Vowel Laxing"
241Ø FOR K=1 TO T
242Ø IF SY(K)=1 AND NA(K)=1 AND BA(K)=-1 AND LO(K)=-1 GOSUB 245Ø
243Ø NEXT K
244Ø GOTO 72Ø
245Ø TN(K)=-1
2451 IF ST(K)=-1 THEN HI(K)=-1
246Ø GOSUB 15ØØØ
247Ø RETURN
25ØØ T$="Palatalization"
251Ø FOR K=1 TO T
2515 IF SG$(K+1)="+" THEN L=2 ELSE L=1
252Ø IF CS(K)=1 AND OB(K)=1 AND CT(K)=1 AND (AN(K)=-1 OR CO(K)=1)
     AND SY(K+L)=1 AND HI(K+L)=1 AND BA(K+L)=-1 GOSUB 255Ø
253Ø NEXT K
254Ø GOTO 72Ø
255Ø AN(K)=-1
2551 CO(K)=1
2552 HI(K)=1
2553 BA(K)=-1
256Ø GOSUB 15ØØØ
257Ø RETURN
26ØØ T$="Glottal Stop Insertion"
261Ø FOR K=1 TO T
262Ø IF SY(K)=1 AND WB(K+1)=1 AND SY(K+2)=1 GOTO 265Ø
263Ø NEXT K
264Ø GOTO 72Ø
265Ø FOR X=T TO K+2 STEP -1
2651 SG$(X+1)=SG$(X)
2652 NEXT X
```

```
 2653 SG$(K+2)="?"
 2654 T=T+1
 266Ø GOSUB 12ØØØ
 267Ø GOTO 261Ø
 999Ø T$="Phonetic Output":PRINT STRING$(G-LEN(T$)," ")+T$;": ";
 9991 FOR X=1 TO T
 9992 IF SG$(X)="+" GOTO 9995
 9993 IF SG$(X)="#" THEN SG$(X)=" "
 9994 PRINT SG$(X);
 9995 NEXT X
 9996 END
11ØØØ FOR K=1 TO T
11Ø1Ø IF SG$(K)="+" AND (SG$(K+1)="+" OR SG$(K+1)="#") GOTO 11Ø4Ø
11Ø2Ø NEXT K
11Ø3Ø GOTO 12ØØØ
11Ø4Ø T=T-1
11Ø5Ø FOR X=K TO T
11Ø6Ø SG$(X)=SG$(X+1)
11Ø7Ø NEXT X
11Ø8Ø GOTO 11ØØØ
12ØØØ FOR K=1 TO T
12Ø1Ø READ A$,SE,WB,CS,OB,SY,CT,AN,CO,VO,HI,LO,BA,RN,TN,NA,ST
12Ø2Ø IF A$<> SG$(K) GOTO 12Ø1Ø
12Ø3Ø SE(K)=SE
12Ø4Ø WB(K)=WB
12Ø5Ø CS(K)=CS
12Ø6Ø OB(K)=OB
12Ø7Ø SY(K)=SY
12Ø8Ø CT(K)=CT
12Ø9Ø AN(K)=AN
121ØØ CO(K)=CO
1211Ø VO(K)=VO
1212Ø HI(K)=HI
1213Ø LO(K)=LO
1214Ø BA(K)=BA
1215Ø RN(K)=RN
1216Ø TN(K)=TN
1217Ø NA(K)=NA
1218Ø ST(K)=ST
1219Ø RESTORE
122ØØ NEXT K
13ØØØ Z$=""
13Ø1Ø FOR X=1 TO T
13Ø2Ø Z$=Z$+SG$(X)
13Ø3Ø NEXT X
13Ø4Ø RETURN
15ØØØ READ A$,SE,WB,CS,OB,SY,CT,AN,CO,VO,HI,LO,BA,RN,TN,NA,ST
15Ø1Ø IF SE<> SE(K) GOTO 15ØØØ
15Ø2Ø IF WB<> WB(K) GOTO 15ØØØ
15Ø3Ø IF ST<> ST(K) GOTO 15ØØØ
15Ø4Ø IF NA<> NA(K) GOTO 15ØØØ
15Ø5Ø IF CS<> CS(K) GOTO 15ØØØ
```

```
15Ø6Ø IF OB<>CT(K)  GOTO 15ØØØ
15Ø7Ø IF SY<>SY(K)  GOTO 15ØØØ
15Ø8Ø IF CT<>CT(K)  GOTO 15ØØØ
15Ø9Ø IF AN<>AN(K)  GOTO 15ØØØ
151ØØ IF CO<>CO(K)  GOTO 15ØØØ
1511Ø IF VO<>VO(K)  GOTO 15ØØØ
1512Ø IF HI<>HI(K)  GOTO 15ØØØ
1513Ø IF LO<>LO(K)  GOTO 15ØØØ
1514Ø IF BA<>BA(K)  GOTO 15ØØØ
1515Ø IF RN<>RN(K)  GOTO 15ØØØ
1516Ø IF TN<>TN(K)  GOTO 15ØØØ
1517Ø SG$(K)=A$
1518Ø RESTORE
1519Ø GOTO 13ØØØ
```

3. Some Sample Runs

The derivations presented in (20)-(28) below demonstrate how the program presented in Section 2 elicits hypothetical underlying representations from the program user and then maps them step by step into their systematic phonetic counterparts through the application of rules (1)-(17) in the linear order (16)-(12)-(1)-(2)-(3)-(4)-(5)-(6)-(7)-(8)-(9)-(10)-(11)-(13)-(14)-(15)-(17).

(20) Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)
to separate each segment & boundary from its neighbors
? *s/i/k/o/+/t/a/b/e/+/SH/u/+/u/n/d/a*

```
 Underlying Representation: ##siko+tabe+SHu+unda##
             Palatalization: ##SHiko+tabe+SHu+unda##
           Alpha-Switching: ##SHiku+tabe+SHy+ynda##
   Obstruent Dissimilation: ----------------------
  High Back Vowel Lowering: ##SHiko+tabe+SHy+ynda##
      Front Vowel Rounding: ##SHiko+tabØ+SHy+ynda##
         Consonant Voicing: ##SHigo+dabØ+ZHy+ynda##
          Stress Placement: ##SHigo+dabØ+ZHy+y´nda##
                h-Deletion: ----------------------
Word-final Obst. Devoicing: ----------------------
   Geminate Simplification: ##SHigo+dabØ+ZHy´+nda##
               Metathesis: ----------------------
        Nasal Assimilation: ----------------------
                i-Umlaut: ----------------------
              Nasalization: ----------------------
           Vowel Insertion: ----------------------
    Nasalized Vowel Laxing: ----------------------
    Glottal Stop Insertion: ----------------------
           Phonetic Output:   SHigodabØZHy´nda
```

(21) Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)
to separate each segment & boundary from its neighbors
? *s/o/t/t/+/r/i/#/s/y/b/o/t/+/r/i/t/i*

```
 Underlying Representation: ##sott+ri#sybot+riti##
             Palatalization: ##sott+ri#SHybot+riti##
           Alpha-Switching: ##sutt+ri#SHybut+riti##
   Obstruent Dissimilation: ##sust+ri#SHybut+riti##
  High Back Vowel Lowering: ----------------------
      Front Vowel Rounding: ----------------------
         Consonant Voicing: ##sust+ri#SHybud+ridi##
          Stress Placement: ##su´st+ri#SHybu´d+ridi##
                h-Deletion: ----------------------
Word-final Obst. Devoicing: ----------------------
   Geminate Simplification: ----------------------
               Metathesis: ----------------------
        Nasal Assimilation: ----------------------
                i-Umlaut: ##su´st+ri#SHyby´d+ridi##
              Nasalization: ----------------------
           Vowel Insertion: ##su´sat+ri#SHyby´d+ridi##
    Nasalized Vowel Laxing: ----------------------
    Glottal Stop Insertion: ----------------------
           Phonetic Output:   su´satri SHyby´dridi
```

(22) Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)
to separate each segment & boundary from its neighbors
? *s/a/m/b/#/k/o/l/+/i/NG/t*

```
 Underlying Representation: ##samb#kol+iNGt##
            Palatalization: ----------------
          Alpha-Switching: ##samb#kul+iNGt##
   Obstruent Dissimilation: ----------------
  High Back Vowel Lowering: ----------------
      Front Vowel Rounding: ----------------
         Consonant Voicing: ----------------
          Stress Placement: ##sa´mb#kul+i´NGt##
                h-Deletion: ----------------
Word-final Obst. Devoicing: ##sa´mp#kul+i´NGt##
   Geminate Simplification: ----------------
               Metathesis: ----------------
        Nasal Assimilation: ##sa´mp#kul+i´nt##
                  i-Umlaut: ##sa´mp#kyl+i´nt##
             Nasalization: ##saM´p#kyl+iM´t##
          Vowel Insertion: ##saM´pa#kyl+iM´t##
    Nasalized Vowel Laxing: ##saM´pa#kyl+IM´t##
    Glottal Stop Insertion: ----------------
           Phonetic Output:   saM´pa kylIM´t
```

(23) Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)
to separate each segment & boundary from its neighbors
? *s/o/f/+/h/e/GH/i/#/a/p/i/n*

```
 Underlying Representation: ##sof+heGHi#apin##
            Palatalization: ##sof+heZHi#apin##
          Alpha-Switching: ##suf+heZHi#apin##
   Obstruent Dissimilation: ##sup+heZHi#apin##
  High Back Vowel Lowering: ----------------
      Front Vowel Rounding: ##sup+heZHi#apyn##
         Consonant Voicing: ##sup+heZHi#abyn##
          Stress Placement: ##su´p+heZHi#aby´n##
                h-Deletion: ##su´p+eZHi#aby´n##
Word-final Obst. Devoicing: ----------
   Geminate Simplification: ----------
               Metathesis: ----------
        Nasal Assimilation: ----------
                  i-Umlaut: ##su´p+eZHi#@by´n##
             Nasalization: ----------
          Vowel Insertion: ----------
    Nasalized Vowel Laxing: ----------
    Glottal Stop Insertion: ##su´p+eZHi#?@by´n##
           Phonetic Output:   su´peZHi ?@by´n
```

(24) Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)
to separate each segment & boundary from its neighbors
? *p/e/r/+/k/o/m/+/k/a/p/+/t/i/d*

```
 Underlying Representation: ##per+kom+kap+tid##
             Palatalization: ------------------
           Alpha-Switching: ##per+kum+kap+tid##
  Obstruent Dissimilation: ##per+kum+kaf+tid##
 High Back Vowel Lowering: ------------------
     Front Vowel Rounding: ##pØr+kum+kaf+tid##
        Consonant Voicing: ##pØr+gum+gaf+tid##
         Stress Placement: ##pØr+gum+gaf+ti´d##
               h-Deletion: --------------------
Word-final Obst. Devoicing: ##pØr+gum+gaf+ti´t##
  Geminate Simplification: --------------------
              Metathesis: ##pØg+rum+gaf+ti´t##
       Nasal Assimilation: ##pØg+ruNG+gaf+ti´t##
                 i-Umlaut: ##pØg+ruNG+g@f+ti´t##
             Nasalization: --------------------
          Vowel Insertion: ##pØg+ruNG+g@fa+ti´t##
  Nasalized Vowel Laxing: ----------------------
  Glottal Stop Insertion: ----------------------
          Phonetic Output:   pØgruNGg@fati´t
```

(25) Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)
to separate each segment & boundary from its neighbors
? *s/i/+/i/#/o/n/d/i/l/+/p/a/d*

```
 Underlying Representation: ##si+i#ondil+pad##
             Palatalization: ##SHi+i#ondil+pad##
           Alpha-Switching: ##SHi+i#undil+pad##
  Obstruent Dissimilation: ------------------
 High Back Vowel Lowering: ------------------
     Front Vowel Rounding: ------------------
        Consonant Voicing: ##SHi+i#undil+bad##
         Stress Placement: ##SHi´+i#undil+ba´d##
               h-Deletion: --------------------
Word-final Obst. Devoicing: ##SHi´+i#undil+ba´t##
  Geminate Simplification: ##SHi´#undil+ba´t##
              Metathesis: ##SHi´#undib+la´t##
       Nasal Assimilation: ------------------
                 i-Umlaut: ##SHi´#yndib+la´t##
             Nasalization: ##SHi´#yMdib+la´t##
          Vowel Insertion: ------------------
  Nasalized Vowel Laxing: ##SHi´#OEMdib+la´t##
  Glottal Stop Insertion: ##SHi´#?OEMdib+la´t##
          Phonetic Output:   SHi´ ?OEMdibla´t
```

(26) Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)
to separate each segment & boundary from its neighbors
? *b/e/#/o/p/i/#/#/a/x/+/x/i/#/OEM/b*

```
 Underlying Representation: ##be#opi##ax+xi#OEMb##
             Palatalization: ##be#opi##ax+SHi#OEMb##
          Alpha-Switching: ##be#upi##ax+SHi#OEMb##
   Obstruent Dissimilation: ##be#upi##ak+SHi#OEMb##
  High Back Vowel Lowering: ----------------------
      Front Vowel Rounding: ##bØ#upy##ak+SHi#OEMb##
         Consonant Voicing: ##bØ#uby##ak+SHi#OEMb##
          Stress Placement: ##bØ´#u´by##a´k+SHi#OEM´b##
                h-Deletion: -------------------------
Word-final Obst. Devoicing: ##bØ´#u´by##a´k+SHi#OEM´p##
   Geminate Simplification: -------------------------
               Metathesis: -------------------------
        Nasal Assimilation: -------------------------
                 i-Umlaut: ##bØ´#y´by##@´k+SHi#OEM´p##
             Nasalization: -------------------------
          Vowel Insertion: ##bØ´#y´by##@´ka+SHi#OEM´p##
    Nasalized Vowel Laxing: ---------------------------
    Glottal Stop Insertion: ##bØ´#?y´by##@´ka+SHi#?OEM´p##
          Phonetic Output:   bØ´ ?y´by  @´kaSHi ?OEM´p
```

(27) Please enter an UNDERLYING REPRESENTATION, using slashes (/'s)
to separate each segment & boundary from its neighbors
? *b/a/d/+/b/i/d/#/b/o/d/#/#/b/e/t/+/o/NG/+/b/i*

```
 Underlying Representation: ##bad+bid#bod##bet+oNG+bi##
             Palatalization: -------------------------
          Alpha-Switching: ##bad+bid#bud##bet+uNG+bi##
   Obstruent Dissimilation: ##baz+bid#bud##bet+uNG+bi##
  High Back Vowel Lowering: -------------------------
      Front Vowel Rounding: ##baz+byd#bud##bØt+uNG+by##
         Consonant Voicing: ##baz+byd#bud##bØd+uNG+by##
          Stress Placement: ##baz+by´d#bu´d##bØd+u´NG+by##
                h-Deletion: ---------------------------
Word-final Obst. Devoicing: ##baz+by´t#bu´t##bØd+u´NG+by##
   Geminate Simplification: ---------------------------
               Metathesis: ---------------------------
        Nasal Assimilation: ##baz+by´t#bu´t##bØd+u´m+by##
                 i-Umlaut: ##b@z+by´t#bu´t##bØd+y´m+by##
             Nasalization: ---------------------------
          Vowel Insertion: ##b@za+by´ta#bu´t##bØd+y´m+by##
    Nasalized Vowel Laxing: ----------------------------
    Glottal Stop Insertion: ---------------------------
          Phonetic Output:   b@zaby´ta bu´t  bØdy´mby
```

(28) Please enter an UNDERLYING REPRESENTATION, using slashes (/´s) to separate each segment & boundary from its neighbors
? *p/!/s/a/s/+/h/i/#/i/p/!/n/a/m/+/t/o/#/ZH/a/b*

Underlying Representation:	##p!sas+hi#ip!nam+to#ZHab##
Palatalization:	----------
Alpha-Switching:	##p!sas+hi#ip!nam+tu#ZHab##
Obstruent Dissimilation:	##p!sat+hi#ip!nam+tu#ZHab##
High Back Vowel Lowering:	##p↑sat+hi#ip!nam+tu#ZHab##
Front Vowel Rounding:	----------
Consonant Voicing:	##p↑zat+hi#ib!nam+du#ZHab##
Stress Placement:	##p↑za´t+hi#ib!na´m+du#ZHa´b##
h-Deletion:	##p↑za´t+i#ib!na´m+du#ZHa´b##
Word-final Obst. Devoicing:	##p↑za´t+i#ib!na´m+du#ZHa´p##
Geminate Simplification:	----------
Metathesis:	----------
Nasal Assimilation:	##p↑za´t+i#ib!na´n+du#ZHa´p##
i-Umlaut:	##p↑z@´t+i#ib!na´n+du#ZHa´p##
Nasalization:	----------
Vowel Insertion:	----------
Nasalized Vowel Laxing:	----------
Glottal Stop Insertion:	##p↑z@´t+i#?ib!na´n+du#ZHa´p##
Phonetic Output:	p↑z@´ti ?ib!na´ndu ZHa´p

4. Some technical notes

The program listed in Section 2 requires approximately 16K of memory. How it operates is outlined below:

LINE(S) FUNCTION

10-172 Preliminaries

20 Reserves 200 bytes of memory space for strings and clears the CRT (or monitor) screen.

30 Defines all non-string variables as integers.

40-60 Dimension 17 arrays so that the strings upon which the phonological rules operate may contain up to 30 segment and boundary elements. SG$(n) will store the symbol (or symbols) corresponding to the n-th segment or boundary in the string, while SE(n), WB(n), CS(n), OB(n), etc. will store the specifications for the features 'segment', 'word boundary', 'consonantal', 'obstruent', etc. associated with that segment or boundary. The +'s, -'s, and blanks of classificatory matrix (19) are replaced in this program by 1's, -1's, and 0's, respectively.

100-172 List each segment and boundary found in (19) together with its specifications for the features SE, WB, CS, OB, SY, CT, AN, CO, VO, HI, LO, BA, RN, TN, NA, ST (in that order).

200-400 Obtaining the first line of the derivation

200 Lets T$ equal the title of the first line of the derivation.

210-230 Elicit a hypothetical underlying representation (UR) from the program user.

240 Adds a pair of #'s to the beginning and end of the UR.

250 Initializes the variables T, A, and G. The most important of these variables is T, for it is used throughout the program to keep track of the number of segments and boundaries found in the utterance at any given stage in the derivation.

260-370 Scan the UR from left to right, assigning each segment or boundary encountered to the variable SG$(T), where T indicates the distance of the segment or boundary in question from the beginning of the UR.

38Ø Runs a subroutine consisting of lines 12ØØØ-13Ø4Ø. This subroutine compares each segment and boundary of the UR with the first element of data lines 1ØØ-172 until it finds a match. It then assigns the feature specifications found on that line to integer variables SE(K), WB(K), CS(K), OB(K), etc. whose subscript K corresponds to that of the string variable SG$(K) associated with the segment or boundary in question. Lines 13ØØØ-13Ø4Ø concatenate all of the segments and boundaries that constitute the UR and then store the resulting string temporarily in the memory location labeled Z$.

39Ø Prints the first line of the derivation, which consists of the title 'Underlying Representation' followed by a colon, a space, and the underlying representation assigned to Z$.

4ØØ Equates W$ with Z$.

7ØØ-267Ø Ordering and applying the rules

7ØØ Begins a loop in which the rules are applied and their outputs printed.

71Ø Specifies the order in which the 17 rules beginning on every hundredth line from 1ØØØ to 26ØØ are applied. It does this by causing the computer to branch, in the desired sequence, to those lines.

1ØØØ-267Ø Contain the 17 phonological rules, each of which operates as follows: It scans the segments and boundaries of string Z$ and performs its structural change upon those substrings of Z$ which meet its structural description. Each time it applies, the rule causes the computer to execute one of the following subroutines:

Subroutine 11ØØØ

Operating after the deletion of a segment, this subroutine eliminates any superfluous morpheme boundaries and then assigns values to the variables SE(K), WB(K), CS(K), OB(K), etc. that characterize the segment or boundary now occupying each 'slot' K in the shortened string.

Subroutine 12ØØØ

Operating after the insertion of a segment, this subroutine assigns values to the variables SE(K), WB(K), CS(K), OB(K), etc. that characterize the segment or boundary now occupying each 'slot' K of the expanded string.

Subroutine 15000

Operating after the structural change of a feature-switching rule, this subroutine compares the feature specifications of the altered segment with those listed on the data lines until it finds a segment with precisely the same feature values as the altered segment. It then assigns the symbol (or symbols) representing that segment to the string variable SG$(K) that corresponds to the altered segment.

Note:
All three subroutines end with lines 13000-13040, which link together all of the segments and boundaries of the modified utterance and assign them to the variable Z$, replacing the string formerly stored there.

720 Prints the title of the rule just processed, lining up the end of that title with the final letter of the title printed on the preceding line of the derivation. A colon and a blank space are printed after the title.

730 Prints the output of the rule just processed immediately after the space left by the print statement in line 720, unless the rule has brought about no change, in which case the rule's output is replaced by a series of hyphens equal in length to the form printed on the preceding line.

740 Equates W$ with Z$.

750 Completes the loop begun on line 700. Since the iteration variable H in line 700 equals '1 to 17', the computer must make seventeen passes through the loop, one for each phonological rule in the program.

760 Causes the computer to jump to line 9990.

9990-9996 Obtaining the last line of the derivation

9990 Prints the first half of the final line of the derivation, which consists of the title 'Phonetic Output', a colon, and a space.

9991 Begins a loop with as many iterations as there are boundaries and segments in the output of the last rule in the ordered set.

9992 Deletes all morpheme boundaries.

9993 Replaces all #'s with blank spaces.

9994 Supplies the second half of the last line of the derivation by printing the phonetic output one segment (or space) at a time.

9995 Completes the loop begun on line 9991.

9996 Terminates the run.

5. Pedagogical Applications

The most obvious pedagogical applications for the program presented in this article lie in the realm of computer-assisted instruction (CAI). Part of any introductory course in phonology must be devoted to teaching students how to read and apply ordered phonological rules. In the classroom, however, the amount of time that can be spent on the development of these basic skills is normally less than optimal. By contrast, a program like that in Section 2 allows students to practice both tasks to their hearts' content. Their computer-assisted work can take the following forms:

(29) a. Seated at a home computer (or the terminal of a larger data processing system) and provided with a hard copy of the rules of the program that has been loaded into the computer's memory, the students can be asked to input different underlying representations and to try to anticipate how they will be affected by each rule before the computer prints the answer on the next line of the ensuing derivation.

b. When the students run out of UR's, they can continue the exercise by making up new ones and inputting them.

c. As a further challenge, the students can be given a list of systematic phonetic representations for which they must attempt to supply all possible underlying representations. They can check their work by typing in their answers and seeing whether the resulting derivations yield the expected surface forms.

d. An exercise of a different sort consists in giving the students a set of UR's and a corresponding set of phonetic forms and asking them to determine the sequence in which the rules must be applied in order to convert the former into the latter. They can easily verify their answers by changing the order of the line numbers listed after the GOTO statement on line 71Ø and then running derivations with the modified program.

The program presented in this article is, of course, only intended as a sample. Teachers can change the number and complexity of the rules contained in it as they see fit in order to provide their pupils with the desired amount of practice.

The educational value of the program in Section 2 is not limited to its CAI applications. In addition to functioning as an extension of the instructor, the program can serve as the object of much valuable experimentation and study. In an age when computers seem destined to play ever expanding roles in virtually all spheres of human endeavor, including linguistic research, conscientious teachers of our subject should seize

every possible opportunity to cultivate their students' programming talents. Phonologists, for example, could teach their students not only how to write rules describing a particular body of phonetic data, but also how to incorporate those rules into simple computer programs for which a program like that in Section 2 might serve as a model. Before attempting to write rule-testing programs of their own, however, the students could 'get their feet wet' by analyzing and modifying the model program. A number of possible assignments come to mind:

(30) a. The students could be quizzed about the functions of various lines in the program.

b. They could be asked to suggest changes in the program that would increase the speed at which its rules are processed.

c. They could be required to add new rules or to modify existing ones. For example, they could be asked to ...

i. restrict the domain of Vowel Insertion (lines 2300-2370 of the program) to the word,

ii. transform the rule of Obstruent Dissimilation (lines 1000-1070) into one of Obstruent Assimilation that makes contiguous obstruents agree in their specifications for the feature 'continuant', or

iii. revise the Metathesis rule (lines 1800-1870) so that it applies only to liquids and obstruents separated by a morpheme boundary.

d. They could be instructed to expand the inventory of segments and boundaries used in the program or to alter the set of distinctive features that define them. They might, for example, be told to add interdental fricatives to the set of segments classified in the data lines.

e. On a more advanced level, they might be assigned the task of restructuring the program to support simultaneous or random-sequential rule application, derivational or surface-phonetic constraints, global rules, diacritic features, or various other theoretical concepts and constructs.

f. Students with a background in computer science might even be given the assignment of translating the BASIC program into another computer language and assessing the relative merits of BASIC and that language as frameworks for phonological description.

Incorporating a component of computer programming into elementary linguistics courses would, in my opinion, have a number of beneficial results. It would better prepare those students who make linguistics their career for the linguistic science of the future. It would benefit those students who do not remain in the field by improving their ability to work with computers, an ability they could put to good use in many different professions. Finally, it would help to dispel the ivory tower image of linguistics and, in so doing, perhaps attract students to our discipline who might otherwise have shied away from it.