University of Arizona, Department of Computer Science

CSc 372 — Assignment 6 — Due noon, Thu Nov 10 — 5%

Christian Collberg

October 27, 2011

1 Introduction

The purpose of this assignment is to get some experience writing a slightly larger Prolog program. We will also become more proficient with pattern-matching, which we have already practiced a bit in the Prolog `l33t` translator.

The following rules apply for this assignment:

1. You may freely introduce auxiliary predicates if that makes your program cleaner.
2. You will be graded primarily on **correctness**, **elegance**, **clarity**, and **documentation**.
3. Every predicate should be commented. At the very least, the comments should state what the predicate does, which arguments it takes, and what result it produces.
4. You may freely use standard Prolog list-manipulation predicates such as `member`, `append`, and `delete`.
5. You should make your predicates as simple and elegant as possible. **You will have points taken off if you use abominations such as the ones below.**
   (a) It's bad Prolog style to use excessive equals (`=`)-operators:
   ```
   % BAD!
   foo(A,B) :- A = B.
   ```
   ```
   % GOOD!
   foo(A,A).
   ```
   (b) Don't use semi-colons for **or**, instead use multiple predicates:
   ```
   % BAD!
   foo(A) :- bar(A); zap(A).
   ```
   ```
   % GOOD!
   foo(A) :- bar(A).
   foo(A) :- zap(A).
   ```
   (c) Don't use the `is`-operator except when you want to evaluate an arithmetic expression:
   ```
   % BAD!
   foo(A,B) :- A is B.
   ```
   ```
   % GOOD!
   foo(A,B) :- A is B+1.
   ```
6. This assignment is graded out of 100. It is worth 5% of your final grade.
7. Input files to the programs in this assignment can be found on the course website.
8. You can work on this assignment in teams of two.

2 Text-to-Speech

In this assignment we are going to build a *text-to-speech* system, a Prolog program which reads out a plain English text. A lot of research has already gone into the production of such systems, which can be seen from the following quote:

> "Speech research is like a huge pit — you can throw any amount of money into it and nothing comes out."
>
> *US federal research funder*

Programs with speech output are particularly useful for individuals who have problems reading text on a screen, such as the visually impaired, dyslexics, or pre-schoolers. Combined with an optical character reader, a text-to-speech system becomes a tool to read out text from books and newspapers, material otherwise inaccessible to those with reading difficulties. The addition of speech of course also enhances the appeal of any computer game!

We will be using a very crude method for converting text to speech, namely stringing together and playing out the phonemes that make up each word. A real system needs to be much more sophisticated in order to produce intelligible speech. Intonation (rising and falling pitch), rhythm, and stress are all important aspects of computer generated speech which we will be ignoring here.

English pronunciation is very difficult. Why, for example, are these five words all pronounced differently: cough, enough, plough, though, through?! The poem *The Chaos* by Dutchman G.N. Trenite (1870–1946) expresses some of the bewilderment of the second language learner:

> Dearest creature in Creation
> Studying English pronunciation,
>   I will teach you in my verse
>   Sounds like corpse, corps, horse and worse.
> It will keep you, Susy, busy,
> Make your head with heat grow dizzy; ...

Nevertheless, it has been shown that a simple rule-based system coupled with a small exception table of common words not covered by the rules will fare remarkably well.

3 Phonetic Rules for English

Section 7 contains about 350 rules which map English words to their phonetic pronunciation. They came out of a project run by the United States Naval Research Laboratory in 1976[1]. The idea is to search the rule set sequentially (from top to bottom) to find a rule which matches the first unmatched letter in the current word. This process is repeated with the next unmatched character. At each step the rules produce one or more English phonemes.

The rules have four major parts: the *left context*, the *match*, the *right context* and the *output*. Here's an example:

[1] The relevant document is *Automatic Translation of English Text to Phonetics by Means of Letter-to-Sound Rules*, by Honey Sue Elovitz, Rodney W. Johnson, McHugh, and Shore, Report 7948 of the United States Naval Research Laboatory, 1976.

The Phoneme codes : Vowels			
Phoneme	Example	Phoneme	Example
IY	bEEt	IH	bIt
EY	gAte	EH	gEt
AE	fAt	AA	fAther
AO	lAWn	OW	lOne
UH	fUll	UW	fOOl
ER	mURdER	AX	About
AH	bUt	AY	hIde
AW	hOW	OY	tOY
YU	YOU		
p	Pack	b	Back
t	Time	d	Dime
k	Coat	g	Goat
f	Fault	v	Vault
TH	eTHer	DH	eiTHer
s	Sue	z	Zoo
SH	leaSH	ZH	leiSure
h	How	m	suM
n	suN	NG	suNG
l	Laugh	w	Wear
y	Young	r	Rate
CH	CHar	j	Jar
WH	WHere		

Table 1: Arpabet phonemes.

```
% Pattern-letter, Pattern-number, LeftContext, MatchChars, RightContext, Phonemes
 pattern(a, 27, [vowel,cons0], [a,l,s], none, [ax,lz]).
```

The match is the letter or letters consumed by the rule. The left and right contexts describe which letters should occur to the left and the right of the matched letters, in order for the rule to be applicable. The output, finally, gives the phonemes corresponding to the matched letters.

For example, the rule `a.27`

```
pattern(a, 27, [vowel,cons0], [a,l,s], none, [ax,lz]).
```

says that the letters `[a,l,s]` could be pronounced as `[ax,lz]` if they appear at the end of a word and are preceded by a vowel and zero or more consonants.

The phonemes are represented by the single lower-case letters or pairs of upper-case letters given in Table 1[2].

Here are some examples:

[2] This transcription system is sometimes referred to as the *arpabet*.

Context Symbols		
Here	**phonemes.pl**	**Explanation**
$\mathcal{V}_+$	`vowel`	One or more vowels (A, O, U, E, I, Y[a]).
$\mathcal{C}_*$	`cons0`	Zero or more consonants (B, C, D, F, G, H, J, K, L, M, N, P, Q, R, S, T, V, W, X, Z).
$\mathcal{C}_1$	`cons1`	One consonant.
$\mathcal{C}_\mathrm{v}$	`consv`	One voiced consonant (B, D, V, G, J, L, M, N, R, W or Z).
$\mathcal{S}$	`suffix`	One of ER, E, ES, ED, ING, ELY (a suffix) (Found in right context only).
$\mathcal{F}$	`front`	One of E, I or Y (a "front" vowel).
$\mathcal{A}$	`any`	Anything.
$\mathcal{N}$	`none`	Nothing.
a–z	`a-z`	Matches the character itself.

[a] Note that in English Y is sometimes a vowel (*try*) and sometimes a consonant (*yellow*).

Table 2: Letter categories used in this document and in `phonemes.pl`.

Word	Phonemes
thimble	/TH,I,m,b,l/
blind	/b,l,AY,n,d/

The left and right context conditions consist of strings of letters or letter-categories, such as vowel and consonant. The rules use the categories given in Table 2.

Here are some examples:

The context	matches the string
$\mathcal{V}_+\mathcal{C}_*$	a
$\mathcal{V}_+\mathcal{C}_*$	all
$\mathcal{V}_+\mathcal{C}_*$	aell
$\mathcal{V}_+\mathcal{C}_1$	ael
$\mathcal{C}_1\mathcal{F}\mathcal{C}_*\mathcal{V}_+$	tire
$\mathcal{C}_1\mathcal{S}$	led
I$\mathcal{C}_1$	is
$\mathcal{V}_+\mathcal{C}_*$CH	orch

Finally, let's look at how the rules are used in matching. Our example word will be asexual:

Step	Rule	Left Context	Match	Right Context	Output
1	A.14	_	a	sexual ($\mathcal{C}_1 = \text{s}, \mathcal{F} = \text{e}, \mathcal{C}_* = x, \mathcal{V}_+ = u$)	/AE/
2	S.9	a ($\mathcal{V}_+ = \text{a}$)	s	exual ($\mathcal{V}_+ = \text{e}$)	/z/
3	E.52	as	e	xual	/EH/
4	X.1	ase	x	ual	/k,s/
5	U.35	asex	u	al	/YUw/
6	A.26	asexu ($\mathcal{V}_+ = \text{u}, \mathcal{C}_* = \text{\_}$)	al	_	/AX,l/

In other words, the rules say that asexual should be pronounced as /AE,z,EH,k,s,YU,AX,l/. Does this coincide with your own intuition?

4 Tasks

In section 7 is a set of rules to translate an English word to the corresponding phonemic representation. Your tasks are as follows:

1. Write predicates `vowel(A)`, `vowelF(A)` (front vowel), `cons(A)` (consonant), `consV(A)` (voiced consonant), and `suffix(A)` to classify letters according to Table 2: [5 points]

```
| ?- vowel(a).
yes
| ?- vowel(b).
no
| ?- vowelF(e).
yes
| ?- vowelF(i).
yes
| ?- vowelF(a).
no
| ?- cons(b).
yes
| ?- cons(c).
yes
| ?- cons(a).
no
| ?- consV(b).
yes
| ?- consV(c).
no
| ?- suffix([i,n,g]).
yes
| ?- suffix([e,d]).
yes
| ?- suffix([a,d]).
no
```

2. The central part of the matching routine is the `prefix_match(Pattern,Chars)` predicate. It essentially implements the rules in Table 2. `prefix_match` matches a list of letter categories (vowel, consontant, etc.) such as found in the rule-set, against a list of letters. First handle the `vowel` category which matches one or more vowels: [10 points]

```
| ?- prefix_match([vowel],[a]).
true ? ;
no
| ?- prefix_match([vowel],[]).
no
| ?- prefix_match([vowel],[x]).
no
| ?- prefix_match([vowel],[a,a,a]).
true ? ;
true ? ;
true ? ;
(1 ms) no
```

Notice that `prefix_match([vowel],[a,a,a]).` succeeds 3 times! This is because `vowel` can match *one or more vowels*. So the first time it succeeds it has matched `[a]`, the second `[a,a]`, the third `[a,a,a]`.

You should, of course, use the `vowel(A)` predicate from the last problem.

3. Now extend `prefix_match` to handle `cons1` which matches *exactly one consonant*: [5 points]

```
| ?- prefix_match([cons1],[b]).
true ? ;
no
| ?- prefix_match([cons1],[b,b]).
true ? ;
no
| ?- prefix_match([cons1],[a,b,b]).
no
| ?- prefix_match([cons1,cons1],[b,b]).
true ? ;
no
| ?- prefix_match([vowel,cons1,cons1],[a,b,b]).
true ? ;
```

Notice that the pattern doesn't have to match *the entire* list of characters, just a prefix of them.

4. Now extend `prefix_match` to handle `consV` which matches *exactly one voiced consonant*: [5 points]

```
| ?- prefix_match([consV],[a,b,b,b]).
no
| ?- prefix_match([consV],[b,b,b]).
true ? ;
no
```

5. Now extend `prefix_match` to handle `front` which matches *exactly one frontal vowel*: [5 points]

```
| ?- prefix_match([front],[a,a]).
no
| ?- prefix_match([front],[e,e]).
true ? ;
no
```

6. Now extend `prefix_match` to handle suffixes: [10 points]

```
| ?- prefix_match([suffix],[i,n,g]).
true ? ;
no
| ?- prefix_match([suffix],[e,d]).
true ? ;
true ? ;
```

The last goal matches twice because both `[e,d]` and `[e]` are defined to be suffixes.

7. Now extend `prefix_match` to handle `cons0` which matches *zero or more consonants* : [20 points]

```
| ?- prefix_match([cons0],[a]).
true ? ;
no
| ?- prefix_match([cons0],[b,a]).
true ? ;
true ? ;
(1 ms) no
| ?- prefix_match([cons0],[b,c,a]).
true ? ;
true ? ;
true ? ;
no
| ?- prefix_match([cons0,cons1,vowel],[b,c,a]).
true ? ;
no
```

8. Write a predicate `suffix_match` which matches from the right rather than from the left: [10 points]

```
| ?- suffix_match([cons1],[b]).
true ? ;
no
| ?- suffix_match([cons1,vowel],[b,a]).
true ? ;
no
| ?- suffix_match([cons1,vowel],[a,b]).
no
| ?- suffix_match([cons1,cons1,vowel],[b,a]).
no
| ?- suffix_match([cons1,cons1,vowel],[c,b,a]).
true ? ;
no
```

HINT: My implementation is 6 lines long.

9. Now you have all the tools ready to write the main pattern matching predicate! `match(Chars, Phonemes, Rules)` takes a list of characters as input and returns a list of phonemes and a list of matching rules as output: [30 points]

```
match(Chars, Phonemes, Rules) :- ...

| ?- match([a,s,e,x,u,a,l],P,R).
P = [ae,z,eh,k,s,yu,w,ax,l]
R = [a:14,s:9,e:52,x:1,u:35,a:26] ? ;
no
| ?- match([t,h,i,m,b,l,e],P,R).
P = [th,ih,m,b,ax,l,silent]
R = [t:17,i:28,m:2,b:7,l:3,e:1] ?
yes
| ?- match([h,e,l,l,o,blank,t,h,e,r,e],P,R).
P = [h,eh,l,silent,aa,pause,dh,er,silent]
R = [h:5,e:52,l:5,l:2,o:48,blank:1,t:7,e:1] ?
```

`match`, of course, uses `prefix_match`, `suffix_match`, and the rules from `phonemes.pl`.

> **HINT: It's a good idea to write an auxiliary predicate (with extra arguments) to make it easier to collect the phonemes and rules. To make things easier you might want to start out by just printing out the phonemes and rules as you find them, and the extend `match` to return them when it works.**

10. You can now try out the top-level `talk` predicates:

```
| ?- readsentence(S).
hello there.
 S = [h,e,l,l,o,blank,t,h,e,r,e] ?

| ?- talk([a,s,e,x,u,a,l],P).
P = [ae,z,eh,k,s,yu,w,ax,l]

| ?- talk.
hello there.
Executing "java play h eh l silent aa pause dh er silent"
```

Note that:

- The files `play.java`, `phonemes.zip`, `match.pl` and `sentence.pl` can be found here: `http://www.cs.arizona.edu/~collberg/Teaching/372/2011/Assignments/index.html`.
- `play.java` will first have to be compiled (`javac play.java` on `lectura`). Its first argument is the file of phonemes and the remaining arguments are the phonemes to play out.
- To actually hear the sounds being played, you need to execute the java program on a machine with a sound card, and speakers! So, `lectura` won't work — try on your home machine instead.
- The file `sentence.pl` contains predicates for reading in a sentence and converting it to a list of characters (atoms):
- `readsentence(S)` assumes that a sentence always ends in a period.
- The `GNU Prolog` predicate `system(S)` calls a shell command from Prolog:

  ```
  | ?- system('ls').
  CVS match.pl phonemes.pl sentence.pl phonemes.zip
  ```

5 Extension A — Numbers [10 EXTRA points]

Extend your program to read out numbers. *"1973"* should, for example, be read as

one-thousand-nine-hundred-and-seventy-three.

- If you do this extension the maximum score you can get is 110 points! You can use this to offset lower scores you got in other assignments. You still can't get more than 40% total for all the assignments in the class.
- You should be able to handle numbers at least up to 9999.
- You don't have to handle negative numbers.
- I am giving you some leeway in how to solve this part of the assignment. You could, for example, translate

  `[s,h,e,blank,w,a,s,blank,o,n,l,y,blank,1,6,o,n,l,y,blank,1,6]`

  into

  ```
  [s,h,e,blank,w,a,s,blank,o,n,l,y,blank,
  s,i,x,t,e,e,n,blank,o,n,l,y,blank,s,i,x,t,e,e,n]
  ```

  and then translate the resulting string as usual. Alternatively, a better-sounding translation could be had by making use of these hard-coded phonetic transcriptions of the number-words:

  > zIHrOW, wAHn, tUW, THrIY, fOWr, fAYv, sIHks, sEHvAXn, EYt, nAYn, tEHn, IYlEHvAXn, twEHlv, THERtIYn, fOWrtIYn, fIHftIYn, sIHkstIYn, sEHvEHntIYn, EYtIYn, nAYntIYn, twEHn-tIY, THERtIY, fAOrtIY, fIHftIY, sIHkstIY, sEHvEHntIY, EYtIY, nAYntIY hAHndrEHd, THAWzA-End, mIHlIYAXn, bIHlIYAXn, AEnd

  A file `numbers.txt` containing these number-words can be found on the website.
- Make sure to describe in your code how your number translation works.

6 Extension B - Iris [10 EXTRA points]

Write a simple question-and-answer system, in the style of Apple's Siri. Call it Iris! Hook your text-to-speech program into this system so that answers are delivered as speech.

7 Submission and Assessment

The deadline for this assignment is noon, Thu Nov 10. It is worth 5% of your final grade.

- You should submit the assignment electronically using `d2l.cs.arizona.edu`.

- You should submit one file, named match.pl.
- Your file should indicate who you are collaborating with, if any.
- Each team only needs to submit one copy of the assignment

Don't show your code to anyone, don't read anyone else's code, don't discuss the details of your code with anyone. If you need help with the assignment see the instructor or the TA.

8 The Rule-Set

Rule	Left Context	Match	Right Context	Output	Example
A.1	$\mathcal{A}$	A	$\mathcal{N}$	AX	
A.2	$\mathcal{N}$	ARE	$\mathcal{N}$	AAr	
A.3	$\mathcal{N}$	AR	O	AXr	
A.4	$\mathcal{A}$	AR	$\mathcal{V}_+$	EHr	
A.5	$\mathcal{C}_1$	AS	$\mathcal{V}_+$	EYs	
A.6	$\mathcal{A}$	A	WA	AX	
A.7	$\mathcal{A}$	AW	$\mathcal{A}$	AO	lawn
A.8	$\mathcal{C}_*$	ANY	$\mathcal{A}$	EHnIY	Delany
A.9	$\mathcal{A}$	A	$\mathcal{C}_1\mathcal{F}\mathcal{V}_+$	EY	
A.10	$\mathcal{V}_+\mathcal{C}_*$	ALLY	$\mathcal{A}$	AXlIY	
A.11	$\mathcal{N}$	AL	$\mathcal{V}_+$	AXl	
A.12	$\mathcal{A}$	AGAIN	$\mathcal{A}$	AXgEHn	
A.13	$\mathcal{V}_+\mathcal{C}_*$	AG	E	IHj	
A.14	$\mathcal{A}$	A	$\mathcal{C}_1\mathcal{F}\mathcal{C}_*\mathcal{V}_+$	AE	
A.15	$\mathcal{C}_*$	A	$\mathcal{C}_1\mathcal{F}$	EY	
A.16	$\mathcal{A}$	A	$\mathcal{C}_1\mathcal{S}$	EY	
A.17	$\mathcal{N}$	ARR	$\mathcal{A}$	AXr	arrive
A.18	$\mathcal{A}$	ARR	$\mathcal{A}$	AEr	carrot
A.19	$\mathcal{C}_*$	AR	$\mathcal{N}$	AAr	tar, star, arm
A.20	$\mathcal{A}$	AR	$\mathcal{N}$	ER	
A.21	$\mathcal{A}$	AR	$\mathcal{A}$	AAr	art
A.22	$\mathcal{A}$	AIR	$\mathcal{A}$	EHr	
A.23	$\mathcal{A}$	AI	$\mathcal{A}$	EY	Daisy
A.24	$\mathcal{A}$	AY	$\mathcal{A}$	EY	play
A.25	$\mathcal{A}$	AU	$\mathcal{A}$	AO	Paul, cauliflower
A.26	$\mathcal{V}_+\mathcal{C}_*$	AL	$\mathcal{N}$	AXl	
A.27	$\mathcal{V}_+\mathcal{C}_*$	ALS	$\mathcal{N}$	AXlz	
A.28	$\mathcal{A}$	ALK	$\mathcal{A}$	AOk	stalker
A.29	$\mathcal{A}$	AL	$\mathcal{C}_1$	AOl	
A.30	$\mathcal{C}_*$	ABLE	$\mathcal{A}$	EYbAXl	
A.31	$\mathcal{A}$	ABLE	$\mathcal{A}$	AXbAXl	
A.32	$\mathcal{A}$	ANG	$\mathcal{F}$	EYnj	
A.33	$\mathcal{A}$	A	$\mathcal{A}$	AE	
B.1	$\mathcal{N}$	BE	$\mathcal{C}_1\mathcal{V}_+$	bIH	
B.2	$\mathcal{A}$	BEING	$\mathcal{A}$	bIYIHNG	
B.3	$\mathcal{N}$	BOTH	$\mathcal{N}$	bOWTH	
B.4	$\mathcal{N}$	BUS	$\mathcal{V}_+$	bIHz	
B.5	$\mathcal{A}$	BUIL	$\mathcal{A}$	bIHl	
B.6	$\mathcal{A}$	B	$\mathcal{A}$	b	

Rule	Left Context	Match	Right Context	Output	Example
C.1	$\mathcal{N}$	CH	$\mathcal{C}_1$	k	
C.2	$\mathcal{C}_1$E	CH	$\mathcal{A}$	k	
C.3	$\mathcal{A}$	CH	$\mathcal{A}$	CH	
C.4	S	CI	$\mathcal{V}_+$	sAY	
C.5	$\mathcal{A}$	CI	A	SH	
C.6	$\mathcal{A}$	CI	O	SH	
C.7	$\mathcal{A}$	CI	EN	SH	
C.8	$\mathcal{A}$	C	$\mathcal{F}$	s	
C.9	$\mathcal{A}$	CK	$\mathcal{A}$	k	
C.10	$\mathcal{A}$	COM	$\mathcal{S}$	kAHm	
C.11	$\mathcal{A}$	C	$\mathcal{A}$	k	
D.1	$\mathcal{V}_+\mathcal{C}_*$	DED	$\mathcal{N}$	dIHd	
D.2	$\mathcal{C}_\mathrm{v}$E	D	$\mathcal{N}$	d	
D.3	$\mathcal{V}_+\mathcal{C}_1\mathcal{C}_*$E	D	$\mathcal{N}$	t	
D.4	$\mathcal{N}$	DE	$\mathcal{C}_1\mathcal{V}_+$	dIH	
D.5	$\mathcal{N}$	DO	$\mathcal{N}$	dUW	
D.6	$\mathcal{N}$	DOES	$\mathcal{A}$	dAHz	
D.7	$\mathcal{N}$	DOING	$\mathcal{A}$	dUWIHNG	
D.8	$\mathcal{N}$	DOW	$\mathcal{A}$	dAW	
D.9	$\mathcal{A}$	DU	A	jUW	
D.10	$\mathcal{A}$	D	$\mathcal{A}$	d	
E.1	$\mathcal{V}_+\mathcal{C}_*$	E	$\mathcal{N}$	Silent	
E.2	'$\mathcal{C}_1\mathcal{C}_*$	E	$\mathcal{N}$	Silent	
E.3	$\mathcal{C}_*$	E	$\mathcal{N}$	IY	
E.4	$\mathcal{V}_+$	ED	$\mathcal{N}$	d	
E.5	$\mathcal{V}_+\mathcal{C}_*$	E	D	Silent	
E.6	$\mathcal{A}$	EV	ER	EHv	
E.7	$\mathcal{A}$	E	$\mathcal{C}_1\mathcal{S}$	IY	
E.8	$\mathcal{A}$	ERI	$\mathcal{V}_+$	IYrIY	
E.9	$\mathcal{A}$	ERI	$\mathcal{A}$	EHrIH	
E.10	$\mathcal{V}_+\mathcal{C}_*$	ER	$\mathcal{V}_+$	ER	
E.11	$\mathcal{A}$	ER	$\mathcal{V}_+$	EHr	
E.12	$\mathcal{A}$	ER	$\mathcal{A}$	ER	
E.13	$\mathcal{N}$	EVEN	$\mathcal{A}$	IYvEHn	
E.14	$\mathcal{V}_+\mathcal{C}_*$	E	W	Silent	
E.15	T	EW	$\mathcal{A}$	UW	
E.16	S	EW	$\mathcal{A}$	UW	
E.17	R	EW	$\mathcal{A}$	UW	
E.18	D	EW	$\mathcal{A}$	UW	
E.19	L	EW	$\mathcal{A}$	UW	
E.20	Z	EW	$\mathcal{A}$	UW	
E.21	N	EW	$\mathcal{A}$	UW	
E.22	J	EW	$\mathcal{A}$	UW	
E.23	TH	EW	$\mathcal{A}$	UW	
E.24	CH	EW	$\mathcal{A}$	UW	
E.25	SH	EW	$\mathcal{A}$	UW	
E.26	$\mathcal{A}$	EW	$\mathcal{A}$	YUw	
E.27	$\mathcal{A}$	E	O	IY	
E.28	$\mathcal{V}_+\mathcal{C}_*$S	ES	$\mathcal{N}$	IHz	
E.29	$\mathcal{V}_+\mathcal{C}_*$C	ES	$\mathcal{N}$	IHz	
E.30	$\mathcal{V}_+\mathcal{C}_*$G	ES	$\mathcal{N}$	IHz	
E.31	$\mathcal{V}_+\mathcal{C}_*$Z	ES	$\mathcal{N}$	IHz	
E.32	$\mathcal{V}_+\mathcal{C}_*$X	ES	$\mathcal{N}$	IHz	
E.33	$\mathcal{V}_+\mathcal{C}_*$J	ES	$\mathcal{N}$	IHz	

Rule	Left Context	Match	Right Context	Output	Example
E.34	$\mathcal{V}_+\mathcal{C}_*$CH	ES	$\mathcal{N}$	IHz	
E.35	$\mathcal{V}_+\mathcal{C}_*$SH	ES	$\mathcal{N}$	IHz	
E.36	$\mathcal{V}_+\mathcal{C}_*$	E	S	Silent	
E.37	$\mathcal{V}_+\mathcal{C}_*$	ELY	$\mathcal{N}$	lIY	
E.38	$\mathcal{V}_+\mathcal{C}_*$	EMENT	$\mathcal{A}$	mEHnt	
E.39	$\mathcal{A}$	EFUL	$\mathcal{A}$	fUHl	
E.40	$\mathcal{A}$	EE	$\mathcal{A}$	IY	
E.41	$\mathcal{A}$	EARN	$\mathcal{A}$	ERn	
E.42	$\mathcal{N}$	EAR	$\mathcal{C}_1$	ER	
E.43	$\mathcal{A}$	EAD	$\mathcal{A}$	EHd	
E.44	$\mathcal{V}_+\mathcal{C}_*$	EA	$\mathcal{N}$	IYAX	
E.45	$\mathcal{A}$	EA	SU	EH	
E.46	$\mathcal{A}$	EA	$\mathcal{A}$	IY	
E.47	$\mathcal{A}$	EIGH	$\mathcal{A}$	EY	
E.48	$\mathcal{A}$	EI	$\mathcal{A}$	IY	
E.49	$\mathcal{N}$	EYE	$\mathcal{A}$	AY	
E.50	$\mathcal{A}$	EY	$\mathcal{A}$	IY	
E.51	$\mathcal{A}$	EU	$\mathcal{A}$	YUw	
E.52	$\mathcal{A}$	E	$\mathcal{A}$	EH	
F.1	$\mathcal{A}$	FUL	$\mathcal{A}$	fUHl	
F.2	$\mathcal{A}$	F	$\mathcal{A}$	f	
G.1	$\mathcal{A}$	GIV	$\mathcal{A}$	gIHv	
G.2	$\mathcal{N}$	G	I$\mathcal{C}_1$	g	
G.3	$\mathcal{A}$	GE	T	gEH	
G.4	SU	GGES	$\mathcal{A}$	gjEHs	
G.5	$\mathcal{A}$	GG	$\mathcal{A}$	g	
G.6	B$\mathcal{V}_+$	G	$\mathcal{A}$	g	
G.7	$\mathcal{A}$	G	$\mathcal{F}$	j	
G.8	$\mathcal{A}$	GREAT	$\mathcal{A}$	grEYt	
G.9	$\mathcal{V}_+$	GH	$\mathcal{A}$	Silent	
G.10	$\mathcal{A}$	G	$\mathcal{A}$	g	
H.1	$\mathcal{N}$	HAV	$\mathcal{A}$	hAEv	
H.2	$\mathcal{N}$	HERE	$\mathcal{A}$	hIYr	
H.3	$\mathcal{N}$	HOUR	$\mathcal{A}$	AWER	
H.4	$\mathcal{A}$	HOW	$\mathcal{A}$	hAW	
H.5	$\mathcal{A}$	H	$\mathcal{V}_+$	h	
H.6	$\mathcal{A}$	H	$\mathcal{A}$	Silent	
I.1	$\mathcal{N}$	IN	$\mathcal{A}$	IHn	
I.2	$\mathcal{N}$	I	$\mathcal{N}$	AY	
I.3	$\mathcal{A}$	IN	D	AYn	
I.4	$\mathcal{A}$	IER	$\mathcal{A}$	IYER	
I.5	$\mathcal{V}_+\mathcal{C}_*$R	IED	$\mathcal{A}$	IYd	
I.6	$\mathcal{A}$	IED	$\mathcal{N}$	AYd	
I.7	$\mathcal{A}$	IEN	$\mathcal{A}$	IYEHn	
I.8	$\mathcal{A}$	IE	T	AYEH	
I.9	$\mathcal{C}_*$	I	$\mathcal{S}$	AY	
I.10	$\mathcal{A}$	I	$\mathcal{S}$	IY	
I.11	$\mathcal{A}$	IE	$\mathcal{A}$	IY	
I.12	$\mathcal{A}$	I	$\mathcal{C}_1\mathcal{F}\mathcal{C}_*\mathcal{V}_+$	IH	
I.13	$\mathcal{A}$	IR	$\mathcal{V}_+$	AYr	
I.14	$\mathcal{A}$	IZ	$\mathcal{S}$	AYz	
I.15	$\mathcal{A}$	IS	$\mathcal{S}$	AYz	
I.16	$\mathcal{A}$	I	D$\mathcal{S}$	AY	
I.17	$\mathcal{F}\mathcal{C}_1$	I	$\mathcal{C}_1\mathcal{F}$	IH	

Rule	Left Context	Match	Right Context	Output	Example
I.18	$\mathcal{A}$	I	T$\mathcal{S}$	AY	
I.19	$\mathcal{V}_+\mathcal{C}_1\mathcal{C}_*$	I	$\mathcal{C}_1\mathcal{F}$	IH	
I.20	$\mathcal{A}$	I	$\mathcal{C}_1\mathcal{F}$	AY	
I.21	$\mathcal{A}$	IR	$\mathcal{A}$	ER	
I.22	$\mathcal{A}$	IGH	$\mathcal{A}$	AY	
I.23	$\mathcal{A}$	ILD	$\mathcal{A}$	AYld	
I.24	$\mathcal{A}$	IGN	$\mathcal{N}$	AYn	
I.25	$\mathcal{A}$	IGN	$\mathcal{C}_1$	AYn	
I.26	$\mathcal{A}$	IGN	$\mathcal{S}$	AYn	
I.27	$\mathcal{A}$	IQUE	$\mathcal{A}$	IYk	
I.28	$\mathcal{A}$	I	$\mathcal{A}$	IH	
J.1	$\mathcal{A}$	J	$\mathcal{A}$	j	
K.1	$\mathcal{N}$	K	N	Silent	
K.2	$\mathcal{A}$	K	$\mathcal{A}$	k	
L.1	$\mathcal{A}$	LO	C$\mathcal{V}_+$	lOW	
L.2	L	L	$\mathcal{A}$	Silent	
L.3	$\mathcal{V}_+\mathcal{C}_1\mathcal{C}_*$	L	$\mathcal{S}$	AXl	
L.4	$\mathcal{A}$	LEAD	$\mathcal{A}$	lIYd	
L.5	$\mathcal{A}$	L	$\mathcal{A}$	l	
M.1	$\mathcal{A}$	MOV	$\mathcal{A}$	mUWv	
M.2	$\mathcal{A}$	M	$\mathcal{A}$	m	
N.1	E	NG	$\mathcal{F}$	nj	
N.2	$\mathcal{A}$	NG	R	NGg	
N.3	$\mathcal{A}$	NG	$\mathcal{V}_+$	NGg	
N.4	$\mathcal{A}$	NGL	$\mathcal{S}$	NGgAXl	
N.5	$\mathcal{A}$	NG	$\mathcal{A}$	NG	
N.6	$\mathcal{A}$	NK	$\mathcal{A}$	NGk	
N.7	$\mathcal{N}$	NOW	$\mathcal{N}$	nAW	
N.8	$\mathcal{A}$	N	$\mathcal{A}$	n	
O.1	$\mathcal{A}$	OF	$\mathcal{N}$	AXv	
O.2	$\mathcal{A}$	OROUGH	$\mathcal{A}$	EROW	
O.3	$\mathcal{V}_+\mathcal{C}_*$	OR	$\mathcal{N}$	ER	
O.4	$\mathcal{V}_+\mathcal{C}_*$	ORS	$\mathcal{N}$	ERz	
O.5	$\mathcal{A}$	OR	$\mathcal{A}$	AOr	
O.6	$\mathcal{N}$	ONE	$\mathcal{A}$	wAHn	
O.7	$\mathcal{A}$	OW	$\mathcal{A}$	OW	
O.8	$\mathcal{N}$	OVER	$\mathcal{A}$	OWvER	
O.9	$\mathcal{A}$	OV	$\mathcal{A}$	AHv	
O.10	$\mathcal{A}$	O	$\mathcal{C}_1\mathcal{S}$	OW	
O.11	$\mathcal{A}$	O	$\mathcal{C}_1$EN	OW	
O.12	$\mathcal{A}$	O	$\mathcal{C}_1$I$\mathcal{V}_+$	OW	
O.13	$\mathcal{A}$	OL	D	OWl	
O.14	$\mathcal{A}$	OUGHT	$\mathcal{A}$	AOt	
O.15	$\mathcal{A}$	OUGH	$\mathcal{A}$	AHf	
O.16	$\mathcal{N}$	OU	$\mathcal{A}$	AW	
O.17	H	OU	S$\mathcal{V}_+$	AW	
O.18	$\mathcal{A}$	OUS	$\mathcal{A}$	AXs	
O.19	$\mathcal{A}$	OUR	$\mathcal{A}$	AOr	
O.20	$\mathcal{A}$	OULD	$\mathcal{A}$	UHd	
O.21	$\mathcal{C}_1$	OU	$\mathcal{C}_1$L	AH	
O.22	$\mathcal{A}$	OUP	$\mathcal{A}$	UWp	
O.23	$\mathcal{A}$	OU	$\mathcal{A}$	AW	
O.24	$\mathcal{A}$	OY	$\mathcal{A}$	OY	
O.25	$\mathcal{A}$	OING	$\mathcal{A}$	OWIHNG	

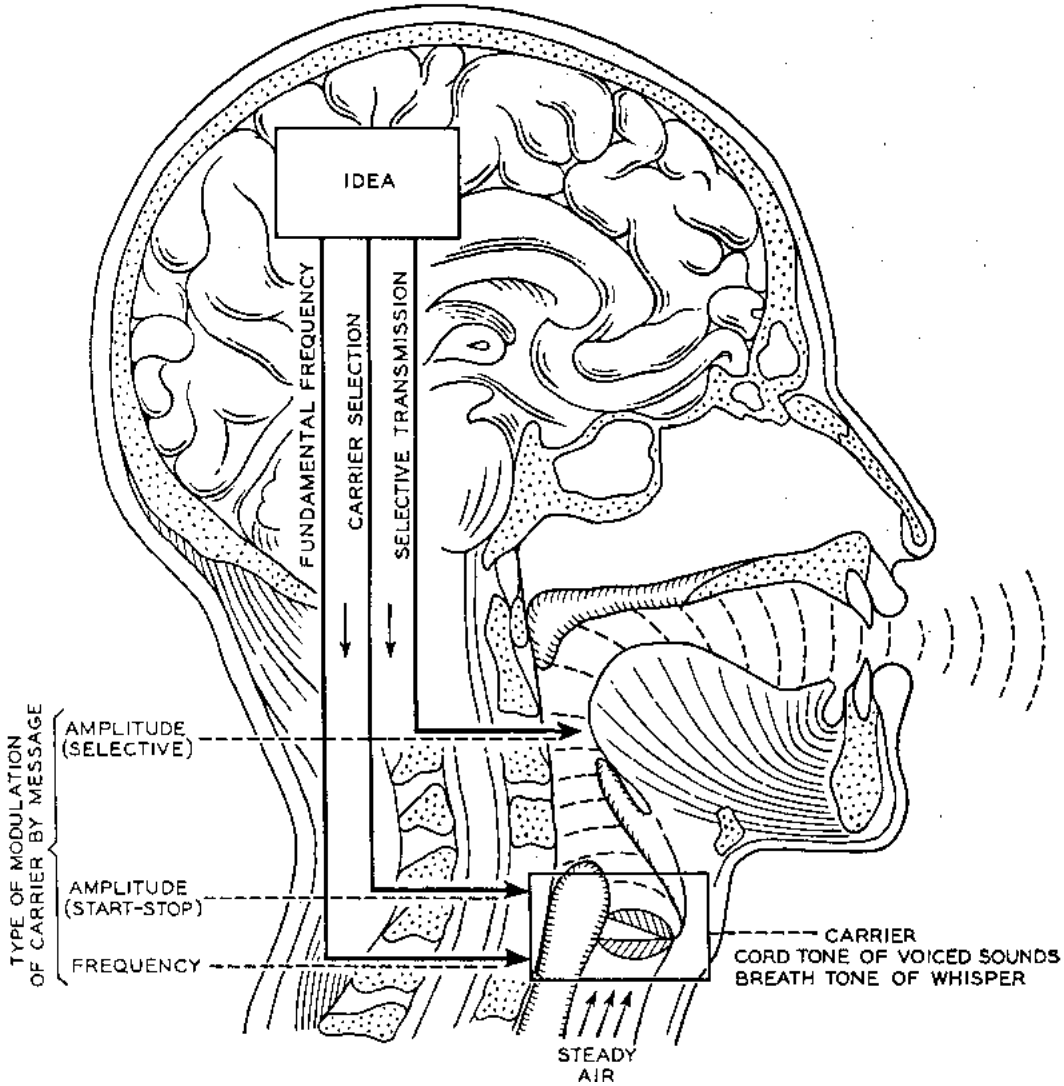
Rule	Left Context	Match	Right Context	Output	Example
O.26	$\mathcal{A}$	OI	$\mathcal{A}$	OY	
O.27	$\mathcal{A}$	OOR	$\mathcal{A}$	AOr	
O.28	$\mathcal{A}$	OOK	$\mathcal{A}$	UHk	
O.29	$\mathcal{A}$	OOD	$\mathcal{A}$	UHd	
O.30	$\mathcal{A}$	OO	$\mathcal{A}$	UW	
O.31	$\mathcal{A}$	O	E	OW	
O.32	$\mathcal{A}$	O	$\mathcal{N}$	OW	
O.33	$\mathcal{A}$	OA	$\mathcal{A}$	OW	
O.34	$\mathcal{N}$	ONLY	$\mathcal{A}$	OWnlIY	
O.35	$\mathcal{N}$	ONCE	$\mathcal{A}$	wAHns	
O.36	$\mathcal{A}$	ON'T	$\mathcal{A}$	OWnt	
O.37	C	O	N	AA	
O.38	$\mathcal{A}$	O	NG	AO	
O.39	$\mathcal{C}_1\mathcal{C}_*$	O	N	AH	
O.40	I	ON	$\mathcal{A}$	AXn	
O.41	$\mathcal{V}_+\mathcal{C}_*$	ON	$\mathcal{N}$	AXn	
O.42	$\mathcal{V}_+\mathcal{C}_1$	ON	$\mathcal{A}$	AXn	
O.43	$\mathcal{A}$	O	ST	OW	
O.44	$\mathcal{A}$	OF	$\mathcal{C}_1$	AOf	
O.45	$\mathcal{A}$	OTHER	$\mathcal{A}$	AHDHER	
O.46	$\mathcal{A}$	OSS	$\mathcal{N}$	AOs	
O.47	$\mathcal{V}_+\mathcal{C}_1\mathcal{C}_*$	OM	$\mathcal{A}$	AHm	
O.48	$\mathcal{A}$	O	$\mathcal{A}$	AA	
P.1	$\mathcal{A}$	PH	$\mathcal{A}$	f	
P.2	$\mathcal{A}$	PEOP	$\mathcal{A}$	pIYp	
P.3	$\mathcal{A}$	POW	$\mathcal{A}$	pAW	
P.4	$\mathcal{A}$	PUT	$\mathcal{N}$	pUHt	
P.5	$\mathcal{A}$	P	$\mathcal{A}$	p	
Q.1	$\mathcal{A}$	QUAR	$\mathcal{A}$	kwAOr	
Q.2	$\mathcal{A}$	QU	$\mathcal{A}$	kw	
Q.3	$\mathcal{A}$	Q	$\mathcal{A}$	k	
R.1	$\mathcal{N}$	RE	$\mathcal{C}_1\mathcal{V}_+$	rIY	
R.2	$\mathcal{A}$	R	$\mathcal{A}$	r	
S.1	$\mathcal{A}$	SH	$\mathcal{A}$	SH	
S.2	$\mathcal{V}_+$	SION	$\mathcal{A}$	ZHAXn	
S.3	$\mathcal{A}$	SOME	$\mathcal{A}$	sAHm	
S.4	$\mathcal{V}_+$	SUR	$\mathcal{V}_+$	ZHER	
S.5	$\mathcal{A}$	SUR	$\mathcal{V}_+$	SHER	
S.6	$\mathcal{V}_+$	SU	$\mathcal{V}_+$	ZHUW	
S.7	$\mathcal{V}_+$	SSU	$\mathcal{V}_+$	SHUW	
S.8	$\mathcal{V}_+$	SED	$\mathcal{N}$	zd	
S.9	$\mathcal{V}_+$	S	$\mathcal{V}_+$	z	
S.10	$\mathcal{A}$	SAID	$\mathcal{A}$	sEHd	
S.11	$\mathcal{C}_1$	SION	$\mathcal{A}$	SHAXn	
S.12	$\mathcal{A}$	S	S	Silent	
S.13	$\mathcal{C}_\mathrm{v}$	S	$\mathcal{N}$	z	
S.14	$\mathcal{V}_+\mathcal{C}_*\mathcal{C}_\mathrm{v}$E	S	$\mathcal{N}$	z	
S.15	$\mathcal{V}_+\mathcal{C}_1\mathcal{C}_*\mathcal{V}_+\mathcal{V}_+$	S	$\mathcal{N}$	z	
S.16	$\mathcal{V}_+\mathcal{C}_1\mathcal{C}_*\mathcal{V}_+$	S	$\mathcal{N}$	s	
S.17	U	S	$\mathcal{N}$	s	
S.18	$\mathcal{C}_*\mathcal{V}_+$	S	$\mathcal{N}$	z	
S.19	$\mathcal{N}$	SCH	$\mathcal{A}$	sk	
S.20	$\mathcal{A}$	S	C$\mathcal{F}$	Silent	
S.21	$\mathcal{V}_+$	SM	$\mathcal{A}$	zm	

Rule	Left Context	Match	Right Context	Output	Example
S.22	$\mathcal{V}_+$	SN	'	zAXn	
S.23	$\mathcal{A}$	S	$\mathcal{A}$	s	
T.1	$\mathcal{N}$	THE	$\mathcal{N}$	DHAX	
T.2	$\mathcal{A}$	TO	$\mathcal{N}$	tUW	
T.3	$\mathcal{A}$	THAT	$\mathcal{N}$	DHAEt	
T.4	$\mathcal{N}$	THIS	$\mathcal{N}$	DHIHs	
T.5	$\mathcal{N}$	THEY	$\mathcal{A}$	DHEY	
T.6	$\mathcal{N}$	THERE	$\mathcal{A}$	DHEHr	
T.7	$\mathcal{A}$	THER	$\mathcal{A}$	DHER	
T.8	$\mathcal{A}$	THEIR	$\mathcal{A}$	DHEHr	
T.9	$\mathcal{N}$	THAN	$\mathcal{N}$	DHAEn	
T.10	$\mathcal{N}$	THEM	$\mathcal{N}$	DHEHm	
T.11	$\mathcal{A}$	THESE	$\mathcal{N}$	DHIYz	
T.12	$\mathcal{N}$	THEN	$\mathcal{A}$	DHEHn	
T.13	$\mathcal{A}$	THROUGH	$\mathcal{A}$	THrUW	
T.14	$\mathcal{A}$	THOSE	$\mathcal{A}$	DHOWz	
T.15	$\mathcal{A}$	THOUGH	$\mathcal{N}$	DHOW	
T.16	$\mathcal{N}$	THUS	$\mathcal{A}$	DHAHs	
T.17	$\mathcal{A}$	TH	$\mathcal{A}$	TH	
T.18	$\mathcal{V}_+\mathcal{C}_*$	TED	$\mathcal{N}$	tIHd	
T.19	S	TI	$\mathcal{V}_+$N	CH	
T.20	$\mathcal{A}$	TI	O	SH	
T.21	$\mathcal{A}$	TI	A	SH	
T.22	$\mathcal{A}$	TIEN	$\mathcal{A}$	SHAXn	
T.23	$\mathcal{A}$	TUR	$\mathcal{V}_+$	CHER	
T.24	$\mathcal{A}$	TU	A	CHUW	
T.25	$\mathcal{N}$	TWO	$\mathcal{A}$	tUW	
T.26	$\mathcal{A}$	T	$\mathcal{A}$	t	
U.1	$\mathcal{N}$	UN	I	yUWn	
U.2	$\mathcal{N}$	UN	$\mathcal{A}$	AHn	
U.3	$\mathcal{N}$	UPON	$\mathcal{A}$	AXpAOn	
U.4	T	UR	$\mathcal{V}_+$	UHr	
U.5	S	UR	$\mathcal{V}_+$	UHr	
U.6	R	UR	$\mathcal{V}_+$	UHr	
U.7	D	UR	$\mathcal{V}_+$	UHr	
U.8	L	UR	$\mathcal{V}_+$	UHr	
U.9	Z	UR	$\mathcal{V}_+$	UHr	
U.10	N	UR	$\mathcal{V}_+$	UHr	
U.11	J	UR	$\mathcal{V}_+$	UHr	
U.12	TH	UR	$\mathcal{V}_+$	UHr	
U.13	CH	UR	$\mathcal{V}_+$	UHr	
U.14	SH	UR	$\mathcal{V}_+$	UHr	
U.15	$\mathcal{A}$	UR	$\mathcal{V}_+$	yUHr	
U.16	$\mathcal{A}$	UR	$\mathcal{A}$	ER	
U.17	$\mathcal{A}$	U	$\mathcal{C}_1$	AH	
U.18	$\mathcal{A}$	U$\mathcal{C}_1\mathcal{C}_1$	$\mathcal{A}$	AH	
U.19	$\mathcal{A}$	UY	$\mathcal{A}$	AY	
U.20	G	U	$\mathcal{V}_+$	Silent	
U.21	G	U	$\mathcal{S}$	Silent	
U.22	G	U	$\mathcal{V}_+$	w	
U.23	$\mathcal{V}_+$N	U	$\mathcal{A}$	YUw	
U.24	T	U	$\mathcal{A}$	UW	
U.25	S	U	$\mathcal{A}$	UW	
U.26	R	U	$\mathcal{A}$	UW	

Rule	Left Context	Match	Right Context	Output	Example
U.27	D	U	$\mathcal{A}$	UW	
U.28	L	U	$\mathcal{A}$	UW	
U.29	Z	U	$\mathcal{A}$	UW	
U.30	N	U	$\mathcal{A}$	UW	
U.31	J	U	$\mathcal{A}$	UW	
U.32	TH	U	$\mathcal{A}$	UW	
U.33	CH	U	$\mathcal{A}$	UW	
U.34	SH	U	$\mathcal{A}$	UW	
U.35	$\mathcal{A}$	U	$\mathcal{A}$	YUw	
V.1	$\mathcal{A}$	VIEW	$\mathcal{A}$	vYUw	
V.2	$\mathcal{A}$	V	$\mathcal{A}$	v	
W.1	$\mathcal{N}$	WERE	$\mathcal{A}$	wER	
W.2	$\mathcal{A}$	WA	S	wAA	
W.3	$\mathcal{A}$	WA	T	wAA	
W.4	$\mathcal{A}$	WERE	$\mathcal{A}$	WHEHr	
W.5	$\mathcal{A}$	WHAT	$\mathcal{A}$	WHAAt	
W.6	$\mathcal{A}$	WHOL	$\mathcal{A}$	hOWl	
W.7	$\mathcal{A}$	WHO	$\mathcal{A}$	hUW	
W.8	$\mathcal{A}$	WH	$\mathcal{A}$	WH	
W.9	$\mathcal{A}$	WAR	$\mathcal{A}$	wAOr	
W.10	$\mathcal{A}$	WOR	$\mathcal{C}_1$	wER	
W.11	$\mathcal{A}$	WR	$\mathcal{A}$	r	
W.12	$\mathcal{A}$	W	$\mathcal{A}$	w	
X.1	$\mathcal{A}$	X	$\mathcal{A}$	ks	
Y.1	$\mathcal{A}$	YOUNG	$\mathcal{A}$	yAHNG	
Y.2	$\mathcal{N}$	YOU	$\mathcal{A}$	yUW	
Y.3	$\mathcal{N}$	YES	$\mathcal{A}$	yEHs	
Y.4	$\mathcal{N}$	Y	$\mathcal{A}$	y	
Y.5	$\mathcal{V}_+\mathcal{C}_1\mathcal{C}_*$	Y	$\mathcal{N}$	IY	
Y.6	$\mathcal{V}_+\mathcal{C}_1\mathcal{C}_*$	Y	I	IY	
Y.7	$\mathcal{C}_*$	Y	$\mathcal{N}$	AY	
Y.8	$\mathcal{C}_*$	Y	$\mathcal{V}_+$	AY	
Y.9	$\mathcal{C}_*$	Y	$\mathcal{C}_1\mathcal{F}\mathcal{C}_*\mathcal{V}_+$	IH	
Y.10	$\mathcal{C}_*$	Y	$\mathcal{C}_1\mathcal{V}_+$	AY	
Y.11	$\mathcal{A}$	Y	$\mathcal{A}$	IH	
Z.1	$\mathcal{A}$	Z	$\mathcal{A}$	z	

From `http://ptolemy.eecs.berkeley.edu/eecs20/speech/voder.html`:

> The technology of speech processing, which includes speech modeling, synthesis, encoding, and recognition, dates back to the parametric techniques introduced by Homer Dudley in the late 1930's and early 1940's. These methods are "parametric" in the sense that they construct a model of the acoustic properties of the human vocal tract, and then analyze speech by determining the values of the parameters of the model. Below is a rendition of the basic model from Dudley's 1940 paper, "The Carrier Nature of Speech," published in the The Bell System Technical Journal.



IDEA
FUNDAMENTAL FREQUENCY
CARRIER SELECTION
SELECTIVE TRANSMISSION
TYPE OF MODULATION OF CARRIER BY MESSAGE
AMPLITUDE (SELECTIVE)
AMPLITUDE (START-STOP)
FREQUENCY
CARRIER
CORD TONE OF VOICED SOUNDS
BREATH TONE OF WHISPER
STEADY AIR