P#1.1: Tonight or Tomorrow?

Code Word	Latin Transcription	Code Word	Latin Transcription
🗎❍❑❒❑	**2moro**	♦❑■♲♦	**tonyt**
♦❑❍❑⌘	**tomoz**	🗎■♓♑♒♦	**2night**
♦❑❍❑❒❑	**tomoro**	🗎■♲♦	**2nyt**
♦❑❍❒⬥	**tomrw**	🗎■🖰	**2n8**
♦❑❍	**tom**	♦❑■🖰	**ton8**
♦❑❍❒♋	**tomra**	♦❑■♲♦⌘	**tonytz**
🗎❍❒❑	**2mro**	🗎■♓♦♏	**2nite**
❍❑❒♋	**mora**	♦❑■♓♦	**tonit**

Now decode the following Tweet (into Standard English):

❒♲♦ ■⬥ ⬥♏ ❒ ♎♏♍❑♎♓■♑ ⬥♓■♑♎♓■♑⌘

ryt nw we r decoding wingdingz

Answer*: Right now we are decoding Wingdings***.**

Explanation:

The idea here is to look at the patterns in which the symbol occurs. For instance, Tomorrow has 3 o's in succession separated by consonants. The task also involves exploiting assumptions such as (a) first letter of the words are never deleted, though they might be substituted with a similar sounding letter of character.

Thinking Beyond:

Can you formalize the strategy you used to solve this problem in the form of an algorithm that will automatically discover the potential variants of a set of valid words from a list of contracted words? Note that like in this problem, the script used for the contracted words might be different from that used for the valid words (transliteration, eh?)

P#1.2 SOUNDEX

(a)

Allaway: A400, *Anderson*: A536, *Ashcombe*: A251, *Buckingham*: B252,

Chapman: C155, *Colquhoun*: C425, *Evans*: E152, *Fairwright*: F623,

Kingscott: K523, *Lewis*: L200, *Littlejohns*: L342, *Stanmore*: S356,

Stubbs: S312, *Tocher*: T260, *Tonks*: T520, *Whytehead*: W330.

(b)

1. Leave the first letter in place.
2. Delete *h* and *w*.
3. Replace all consonant letters with digits (letters whose most common sounds are similar are grouped together):

*bpv(f):*1 *cgjkqs(xz):2* *dt:3* *l:4* *mn:5* *r:6*

4. Reduce any sequence of two or more identical digits to a single digit.
5. Delete all vowels (*a*, *e*, *i*, *o*, *u*, *y*).
6. Leave only the first three digits or add zeroes on the right to make the code one letter and three digits long.

(c)

Ferguson: F622, *Fitzgerald*: F326, *Hamnett*: H530, *Keefe*: K100,

Maxwell: M240, *Razey*: R200, *Shaw*: S000, *Upfield*: U143.

Thinking Beyond:

1. What do you think is the rationale behind this encoding? More specifically:
    a. Why the first letter remains intact?
    b. Why the designer would have chosen to have a code-length of 4 (i.e., one char + 3 more digits), instead of 2 or 10?
2. SOUNDEX uses only 6 codes for the letters, and 0 for padding. What are the advantages or disadvantages of extending the codes to 9 (i.e., divide the consonants to 9 classes instead of 6), or 3? If you were to design letter classes with 9 or 3 sets, how would you do it?
3. Try to design a SOUNDEX like encoding scheme for your mother tongue (or any Indian language other than English).

P#2.2: Māori Loanwords

hāma = hammer	*māti* =match-stick	*raina* =line	*tīhi* = cheese
hāpa = harp	*paipa* = pipe	*taraka* =truck	*tūru* = stool
hū =shoe	*piriti* =bridge	*terewhono* =telephone	*wāna* = swan
hūtu =suit	*pūnu* =spoon	*tiā* = jar	*whurutu* =fruit
iniki =ink	*pūtu* = boot	*tiaka* = jug	*wūru* = wool

hekeretari = secretary, pirinihehe = princess, pirihimana = policeman, tiati = judge

Iharaira = Israel, Kiupa= Cuba, Peina = Spain, Tiamani= Germany, Tiapana = Japan

beef = *pīwhi*, bull = p*ū*ru, cart = k*ā*ta, clock = karaka, lease = r*ī*hi, meat = m*īti*, seal = h*īri*, street = t*iriti*, time = taima, watch = w*ā*ta

Explanation:

Maori allows only CV (Consonant-Vowel) syllables. Therefore, Loan words ending with a consonant (e.g., swan) needs a vowel, and similarly, loan words that have multiple consonants (i.e., CCV) needs to be broken down into two syllables by inserting a vowel. E.g., truck = taraka

Marori doesn't have consonants such as b (replaced by p), sh (replaced by h), l (replaced by r), j (replaced by t).

In general, all languages have certain constraints on what combination of consonants and vowels are allowed in a valid word structure. As a result, transliteration of words from a foreign language always implies certain amount of phonological transformations. Usually, these rules are predictable.

Thinking Beyond:

1. Can you write down an algorithm to convert an English loan word to Maori?
2. Imagine that you are transliterating Hindi loan words to English and vice versa. What are the sounds or sound combinations allowed in Hindi that are not allowed in English? What would be their most natural and closest transliteration into English? (E.g., English doesn't have the sound "d" as in "badA" [big]. How does this sound gets transliterated into English?). Repeat the same exercise in the reverse direction. For instance, Hindi doesn't have the English vowel "o" in "hot". [If you do not know Hindi, do the same exercise of any Indian language that you know].

P#2.2 Transliterating Lepcha

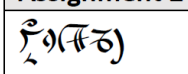
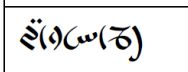
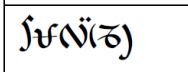
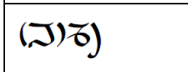
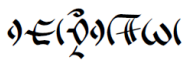
Like all Brahmi derived scripts, Lepcha is an Abugida and is written from left to right.

- The CV (Consonant-vowel) syllables are written by writing down the consonant and the V is marked by a diacritic. E.g., ᰪ (r) + ◌ᰤ (aa) = ᰪᰤ (raa) in Rai (the a here is a long aa like the a in "art").
    - Some diacritics are written before the consonant (e.g., ᰣ (l) + ◌ᰦ (i) = ᰦᰣ (li))
    - Long vowels are represented by the diacritic (◌ᰵ). But certain long vowels, like *oo* has special symbols (◌ᰶ which is essentially (◌ᰵ (therefore, ᰶ is short o)
    - The vowel short *a* (the sound of u in *fun*) is the default sound and is not represented by any diacritic. E.g., Magar = ᰝ (ma) + ᰜᰵ (gar)
- CVC syllables are expressed as the CV syllables along with a diacritic for the last consonant. E.g., ren = ᰪ (r) + ◌ᰧ (e) + ◌ᰲ (n) = ᰪᰧᰲ This is different from many other Brahmi derived scripts where the last consonant is attached to the next syllable. Only some consonants can occur in the syllable final position: *m, n, ng, r, p, t* etc.
- The diacritic for the syllable-final ng is an exception because it precedes the initial consonant. Thus, *mang* is written as ᰋ (ng) + ᰝ (m) + ◌ᰤ (aa) =ᰋᰝᰤ
- CCV syllables, where the only allowable consonants in the second position are r, y, or l (the problem provides example of only *r* in **Drendzongke**), are written with special diacritics as well. E.g., ᰟ (d) + ◌ᰫ (r) + ◌ᰧ (e) + ◌ᰲ (n) = ᰟᰫᰧᰲ (dren)
- V syllables are written with a dummy consonant (ᰚ) marker with the vowel diacritic. e.g., *i* = ᰦᰚ

The Lepcha consonants used in this problem are: n = ᰣ, p =ᰡ, l = ᰣ, ch = ᰐ, d = ᰟ, dz (j) =ᰖ, k=ᰑ, t=ᰜ, m =ᰝ, b =ᰚ, w =ᰕ, r=ᰪ, g=ᰃ, s=ᰎ, sh=ᰍ, h= ᰦ, dummy (used for vowels) = ᰚ

The Lepcha vowel diacritics used in the problem are: aa=◌ᰤ, i=◌ᰦ, u=◌ᰫ, e=◌ᰧ, oo=◌ᰶ

The Lepcha consonant diacritics (i.e., syllable final consonants) used in this problem are: p=◌ᰬ, n=◌ᰲ, ng=ᰋ◌, m = ◌ᰳ, r=◌ᰵ, t= ◌᰷,

Assignment 1	**Answer**
Language Name	SIKKIMESE
Assignment 2	
	Renjoongmu or Rendzoongmu
	Taamsaangmu
	Hilaammu
	Promu
Assignment 3 Transcribe *Kangchenjunga*	

Explanation:

Unlike English, which is an *alphabetic script*, Lepcha is a more complex script that is called a syllabary. Each syllable is written using a complex composition of the characters. Japanese is another example of syllabary. Other Indic scripts, such as Devanagari, is close to a syllabary, but have some small differences – such as the consonant sound at the end of the syllable (technically called coda) is shown as a part of the next akshara (e.g., the Hindi word **काल** has two akshars, but only one syllable phonologically /kal/. So an akshara is not an exact equivalent of a syllable.

Transliteration strongly depends on the type of scripts involved during the process. An important question one must consider is what should be treated as the minimal unit of the writing system (technically such a minimal unit is called a grapheme). While we consider letters as the minimal unit, but that is only useful in the case of alphabetic scripts. For other kinds of scripts, one must consider syllables or akshars as possible minimal units.

Thinking Beyond:

Read more about the different kinds of writing systems (https://en.wikipedia.org/wiki/Writing_system) and think about what would be the ideal choice for the minimal unit while transliterating between two different kinds of scripts.

P#2.3 Switching *ya* Mixing?

2.1a Dude I think u should try again caz ye [this] tera [your] fault nahi [not] hai [is]. ye [this] CBSE walo [people] ki [of] fault hai [is].

Translation: Dude I think you should try again because it is not your fault at all. This is CBSE folk's fault.

Answer: Switching & mixing*

Dude I think u should try again caz: [Matrix=En]

ye tera fault* nahi hai. [Matrix = Hi]

ye CBSE walo ki fault* hai. [Matrix = Hi]

2.1b Corruption to [grammatical particle] every level pe [at] hai [is] and its complete eradication possible nahin [is not].

Translation: Corruption is at every level and its complete eradication is not possible.

Answer: Mixing, Matrix = Hi

2.1c I had told you exams me [in] difficult questions aayenge [will come].

Translation: I had told you that there will be difficult questions in the exam.

Answer: Switching & mixing*

I had told you [Matrix = En]

Exams* me difficult* questions* aayenge. [Matrix = Hi]

Thinking Beyond:

Can you design a system to automatically classify a mixed language text into code-mixing or code-switching? What resources would you need?

What about automatic identification of the matrix?