Exun 2018

Powered by

UniRely
College Counseling

The Turing Test

Preliminary Round

Instructions To Candidates

1. The Turing Test prelims will be held on 8 October, Monday, 5:10 - 9:10 PM IST.
2. Participants will be able to access the question paper at exun.co/18/turingtest. There will be 3 questions, with four hours to submit the solutions. The points scored for each problem will be mentioned on the question paper itself.
3. Participants are required to submit their responses through Google Docs, OR by writing solutions and uploading clear images/scans of these to Google Drive, and sharing the same with exun@dpsrkp.net. We will be accepting answers in the form of Word Documents, PDF Files, and Google Docs. Please ensure that your solutions reach us by 9:10 PM IST positively, and we recommend submitting before the deadline in order to ensure that we get the files on time.
4. In case of any doubts or clarifications, email exun@dpsrkp.net regarding the same.
5. Ensure that each answer is accompanied by a substantive explanation of the linguistic phenomenon that is taking place. Well-explained solutions will be awarded more points as compared to simply giving the answers themselves. Note that we do not require you to explain how you arrived at the answer. For example in English, we require you to explain a rule as: *plurals of some words are made by suffixing -s to the singular form*. We do not need you to tell us that you found the rule from looking at the plural forms of the nouns *apple*, *banana*, etc. In short, simply tell us your conclusions, and the rules you discovered to help you arrive at the conclusions, not the examples that demonstrate these rules.
6. We recommend that you work these problems out on paper. It helps.
7. Any changes made during the contest will be marked with red. While the problems have been thoroughly tested and the chances of any mistakes are low, we may consider awarding small bounties for finding bugs in the problems. Email these at exun@dpsrkp.net
8. Use of the internet will rarely be of help, but is allowed.
9. Please note that if you send us your solutions as a Word Document, or a PDF file, we will only accept those files which reach us before 9:10 PM. If you share the solutions with us as a Google Docs document, then we will only consider the version that existed at 9:10 PM, and subsequent edits will be ignored. We encourage participants to submit before time, and not to edit or change their solutions once submitted (especially for those submitting through Google Docs).
10. There is NO NEGATIVE MARKING, feel free to guess!

GOOD LUCK!

Problem 1: Graph Theory (20 Points)

Shashwat Goel

As a hobby, some people collect stamps, while others collect coins. Shash has a rather unusual hobby: he is obsessed with graphs, and often buys them at auctions, often finding shortest paths from one graph auction to another. At Exun, he recently procured a mysterious graph with 16 words written beside it, which he is unable to decipher.

Help him with his task of deciphering what each vertex of the graph represents (note that vertices are numbered starting from 0). Your solution should describe how the connection between the graph and the list of words is made, as well as why the words fit to the particular vertices in the graphs, and how the words are related to each other.

- Bird
- Milk
- Colour
- Eagle
- Cow
- Golden
- Clownfish
- Hoopoe
- Animal
- Orange
- Food
- Fish
- Tiger
- Mammal
- Human
- Infant

Note that the problem may seem cryptic in it's own sense, and is meant to give you a first-hand experience of the work involved in the field linguistics.

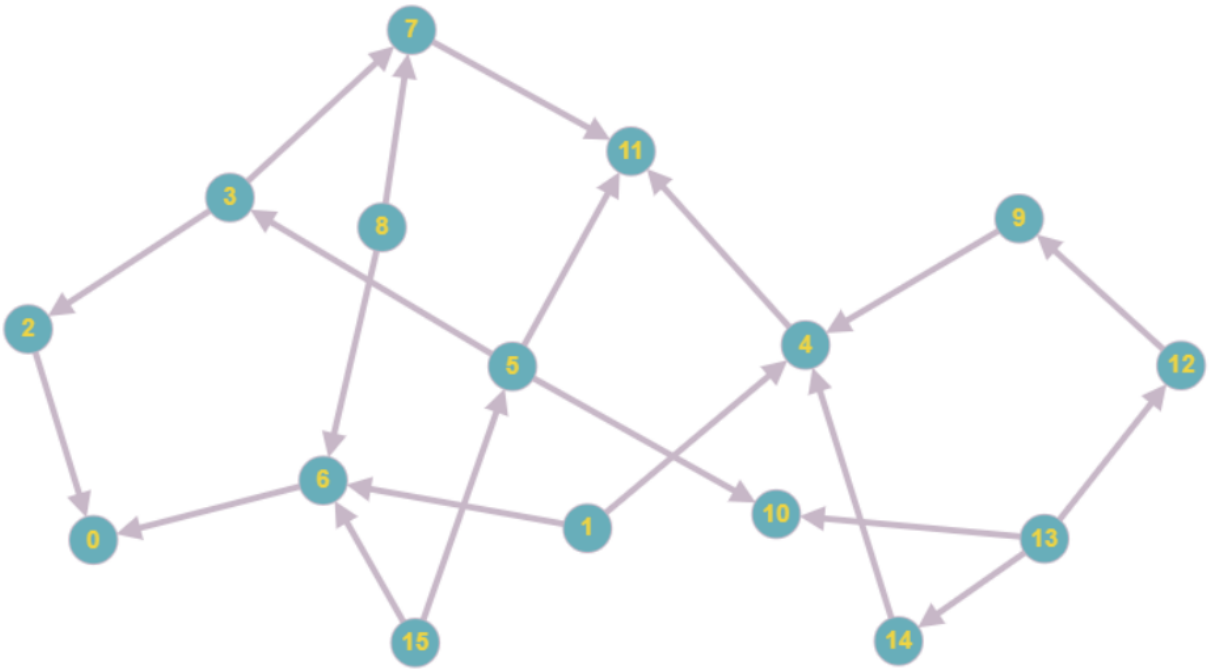


Figure 1: Shash's Graph

Each correctly identified vertex will earn you 1 point, and 4 points will be awarded for correctly explaining the relations between the vertices.

Problem 2: The Crude and the Corrupt (20 Points)

Nikunj Taneja

Originally, Exun was going to demonstrate a sentient AI robot that could converse with participants, designed by Sago. However, Sago had created a rather ingenious system for participants to communicate with the robot. Seeing that English was too difficult to be parsed by the AI, Sago had created his own intermediate level constructed language in which the AI was programmed, which could then be compiled into proper English understandable by participants at Exun. However, Sago woke up today to see that the AI had somehow been corrupted by feeding it wrong data, and he needs your help to restore the AI robot so that it can be shown at Exun.

PART I

Artificial Intelligence relies on the machine learning algorithms identifying patterns from the training dataset, and then correcting them to give more accurate translation. Hence, fixing a machine's errors in translation is an important part of training the AI. In the first part of the question, there is a table comprising of a stemmer in the constructed language. The verb is taken as input, and the computer has generated a crude stem from the data, which has been refined by Sago to show the correct, or actual verb stem of the verbs in the constructed language. However, Sago is tired of manually editing all the verbs according to a given set of rules, so he wants you to analyse the portion of verbs he has already edited, and write the algorithm that another computer can use to show the correct stem, and send the feedback to the AI.

Stemming is the process of reducing inflected (or sometimes derived) words to their word stem, base or root form — generally a written word form. The stem need not be identical to the morphological root of the word; it is usually sufficient that related words map to the same stem, even if this stem is not in itself a valid root. Algorithms for stemming have been studied in computer science since the 1960s. A computer program or subroutine that stems word may be called a stemming program, stemming algorithm, or stemmer. Your task is to figure out the stemming algorithm, using the given data, and figure out the rules for creating the verb stems. Keep in mind that the verb stems here can be derived from the crude stems (which is why Sago is fixing the AI's errors!), and hence, the algorithm follows rules to create a crude stem, and from the crude stem, it creates a verb stem.

Keep in mind that the rules are applied in hierarchical order, that is, Rule I is applied before Rule II, and so on. Note that once a Rule's turn has crossed, it cannot be applied, i.e. Rule I cannot be applied after Rule III, even if a situation arises where it is applicable.

Also, in some verbs, the AI was unable to form a crude stem, so that cell has been left blank. Yet, Sago still has a rule for forming the actual verb stem, even if the machine could not form the crude stem, and you would need to find this rule out as well. To make matters worse, **multiple hierarchies** are possible, because some rules apply in some cases, whereas others do not. All you need to find is a set of rules, and a hierarchy that satisfies *all* the examples.

Infinitive	**Crude Stem**	**Stem**
baggel	bagg	bag
dissel	diss	dis
gogel	gog	goog
ledel	led	leed
mutvel	mutv	mutf
yetel	yet	yeet
nevel	nev	neef
nobel	nob	noob
nozel	noz	noos
nuggel	nugg	nug
oniël	-	onie
pevel	pev	peef
polzel	polz	pols
ptegel	pteg	pteg
sgiël	-	sgie
srobbel	srobb	srob
tuziël	-	tuzie
terel	ter	teer
vannel	vann	van

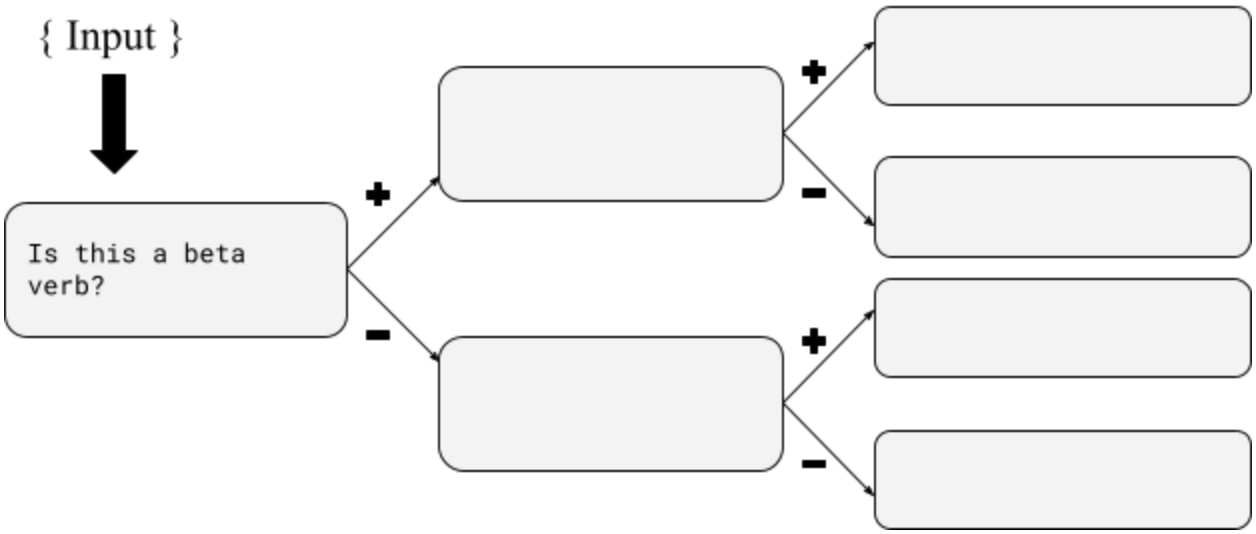
vutel	vut	vuut
xefvel	xefv	xeff
xellel	xell	xel
xerel	xer	xeer
xetvel	xetv	xetf
xuivel	xuiv	xuif
zotel	zot	zoot

PART II

There are two classes of verbs in Sago's constructed language - *alpha* and *beta*. It turns out that the AI is now incorrectly converting verbs into the past tense as well, so Sago needs to fix that dataset as well. Past tense of *alpha* and *beta* verbs are formed differently but the process of past tense formation can be caught in a simple decision tree, which first **converts a verb into its actual verb stem (not the crude stem)**, and then uses the decision tree to create its past tense. The task is to complete the decision tree using the stemming algorithm and the data provided below. Note that ***ü*** is a vowel.

Infinitive	**Past Tense**	**Class**
dümel	deem	beta
glübel	gleeb	beta
gnuivel	gnoof	beta
gtappebel	gtappebme	alpha
hatgel	hatgme	alpha
hüsel	hees	beta
jnassifijetel	jnassifijetre	alpha
missel	misre	alpha
mtokel	mtokre	alpha
niekel	nook	beta
patfel	patfre	alpha
pefteigel	pefteigme	alpha

pemtiekel	pemtook	beta
pezxügel	pezxeeg	beta
piemel	poom	beta
qumiredel	qumiredme	alpha
sbtuirel	sbtoor	beta
sjhaarsel	sjhaarsre	alpha
srtümel	srteem	beta
vetmtierel	vetmtoor	beta
vetmxülel	vetmxeel	beta
zxükel	zxeek	beta



PART I is for 12 points, which include 10 Points for the rules, and 2 Points for the correct hierarchy, or order of the rules. Note that more concise rules (without missing any cases) will lead to higher points. PART II is for 8 Points, which include 1 Point for each rule, and 2 Points for correctly assigning each rule to each box in the above diagram.

Problem 3: Searching for Friends (20 Points)

Sagnik Anupam

As Sago is a bit old-fashioned, he prefers to keep a database of all the amazing linguists and computer scientists he is in contact with, just like a telephone book. However, because he is in Exun, he prefers to keep this in a paperless format, hence, he uses an online database to store all these details. Whenever he wishes to contact someone, instead of flipping the pages of a telephone book, he uses an advanced search function to find the contact details of that person.

For search functions, it is incredibly difficult to parse through large amount of texts and attempt to locate entries. Hence, in information retrieval, people are always looking at ways to shorten the data that they store, in an effort to make retrieval as efficient as possible. When it comes to storing names of people, several algorithms were devised to encode these names efficiently. However, as names vary from region to region, different codes had to be devised for encoding names in a database based on region-centric differences. As these algorithms shortening the names using encoding techniques, it is possible that two different names might end up having the same code. But certain cultures may accept the grouping of two such names as similar, whereas others might reject this grouping, and choose to create a new code. As a result, there are several variations of such algorithms being used.

Given below are 12 names in English, and 11 codes in jumbled order (yes, there is one code that is the same for two different names!).

Haafiz, Nabeel, Zaffar, Mesyurat, Cuci, Abdulla, Omaar, Maqbul, Nargis, Fakir, Taqdir, Nagpaak

376891, 525710, 459100, 931000, 641000, 324100, 847600, 312760, 427100, 696700, 347100

Note: *c* is a consonant pronounced as *ch* in *church*.

Assignment 1: Match the codes to the names, keeping in mind that one of these codes matches to two names.

Assignment 2: Encode the following names using the same encoding system:

1. Istaq
2. Salman
3. Mohammad
4. Aftab
5. Afreen

Assignment 3: Explain the algorithm, and how it encodes the names of people.

Assignment 1 is for 6 Points (0.5 for each correct pairing), and Assignment 2 is for 5 Points. 9 Points will be awarded for an accurate description of all steps of the algorithm in Assignment 3.

END OF PAPER

Don't forget to send your solutions to exun@dpsrkp.net!