ZERO IQ AI

NY Times Mini Crossword Puzzle Alternative Clue Generator

CS 461 – Artificial Intelligence

Final Report

Group Members:

Ege Aydın
Denizhan Soydaş
Ali Özer
Onur Kırmızı
Sina Şahan

Abstract

In this project, we generate alternative clues for the NY Times mini crossword puzzle with an A.I. based program. After obtaining the content of the crossword puzzle from the New York Times's website, our program creates new clues by following the steps of our algorithm. In the end, there are new phrases that lead to the answer of the original crossword puzzle. The clue generation algorithm is created by observing the previous clues. There was not enough puzzle data to use a machine learning algorithm; therefore, pre-trained neural networks for natural language processing (NLP) are used instead. The necessary data about the answers of the puzzle is scraped from the web in forms of definitions and example sentences. Merriam Webster and Word Hippo websites and Google Knowledge Graph API are used for this purpose. The scraped data is then categorized and evaluated. The evaluation process uses Google's pre-trained BERT neural network and the Word2Vec tool. After the evaluation, the most meaningful sentence from our data is selected as the new clue.

Table of Contents

Introduction

In this project, the goal is to write new clues for the NY Times crossword puzzle. For this goal, there were few approaches possible that could be taken. One approach would be using machine learning to analyze years of old clues for given answers and use the generated model to generate new clues. However, there is not enough data to train such a model, and even if there were, the sentence forming capabilities of even cutting-edge AI is not sufficient. To give an example, even OpenAI's new language model called GPT-2, which is said to have "achieved state-of-the-art results," is having trouble creating meaningful sentences (You can try GPT-2 at https://talktotransformer.com/) [1]. For this reason, it was not seen as a feasible solution to use a less sophisticated system, such as the N-gram language model. Therefore, it is decided not to create a sentence from scratch. Instead, selecting a sentence from a given data is a better approach to take. That data is mainly acquired by Merriam-Webster API, Google Knowledge Graph API, Datamuse API, and scraping the web to get example sentences.

Implementation

In order to create new clues for a crossword puzzle, first, it is necessary to know what is accepted as a clue. From puzzles of Joel Fagliano, it is observed that there are five types of clues:

1) Clues that give an example of the answer word’s usage

    Lira: Dollars : U.S. :: ___ : Turkey

2) Clues that defines the answer

    Puma: Mountain lion

3) Clues that give a common knowledge about the answer

    Ugly: Like some Christmas sweaters

4) Clues that show the answer is part of another word

    Dys: Prefix with -lexia

5) Clues that give a rhyming word

    Hint: It rhymes with mint for this answer

Moreover, in the algorithm, five stages are used corresponding to these clue types. For the first type, some example sentences are scraped from the web and evaluated using an NLP neural network called BERT. For the second type, definitions of the answer are acquired from Merriam-Webster, and the best definition is selected using an evaluation algorithm using the Word2Vec technique. For the third type, a tool called Google Knowledge Graph (GKG) is used. The output of the GKG is transformed into sentences and evaluated using the same method when evaluating the definitions. The fourth type is generated by simply using string operations on one-word examples. The last type is acquired sing Datamuse API. One clue is selected among all those clues according to a simple control flow that is described later. Finally, the data is transferred to the non-interactive GUI, which just draws the puzzle grid and shows both the old (scraped from https://www.nytimes.com/crosswords/game/mini) and new clues.

Used codes and API's list

- BeautifulSoup and Selenium libraries for scraping
- NLTK (Natural Language Toolkit) library
- Gensim's Word2Vec library
- Google's Word2Vec data (GoogleNews-vectors-negative300.bin)
- NumPy library
- Google's BERT for example sentence evaluation
    - PyTorch library for neural networking
    - Pytorch_pretrained_bert library for pretrained BERT data
    - Part of the code retrieved from https://huggingface.co/transformers/v1.0.0/quickstart.html
- Google Knowledge Graph API and its Python wrapper retrieved from https://developers.google.com/knowledge-graph
- Merriam-Webster API and its Python wrapper retrieved from https://github.com/fluencer/dictionary-cli-app/blob/master/merriam.py
- WordHippo Website for example sentences
- Datamuse API

1. Example Sentence Evaluation using BERT

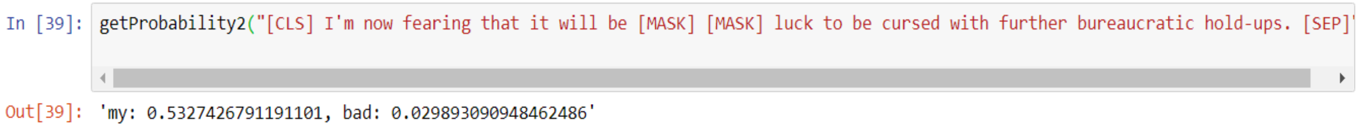
The example sentences are scraped from wordhippo.com, which is basically an example sentence database. However, this site does not always give very stable sentences, such that the sentences may be grammatically incorrect or can even be not in English. Another problem is that the usage of a word may not be fully guessable in every sentence. For example, in the clue "I bought a _____ car," it is quite hard to guess the blank since it may be multiple things. Therefore, there is a need to evaluate and select the most guessable sentence when the answer is removed. For this purpose, a pre-trained neural network called BERT is used.

BERT (Bidirectional Encoder Representations from Transformers) is an open-source neural network-based AI created by Google that can predict a word in a sentence. This ability is necessary to evaluate an example sentence's usability as a clue. The logic is as follows: There exists an example sentence which includes the answer at least once. Mask the answer word and make BERT try to guess what is there in the masked location. Take the probability of the predicted word being the answer. If BERT could guess the word with high probability, most likely, humans can guess the word as well. Unlike other NLP prediction solutions such as the N-gram language model and GPT2, BERT is bidirectional. That means it requires words at both sides of the masked word to operate [1]. This property of it is especially important for our purpose since the answer word would most likely be in the middle of the example sentence rather than the ends of it, therefore only BERT would be able to make a satisfactory prediction.

For example, assume that the example sentence for the answer word "red" is "a shiny red fire truck." First, we put necessary tags and mask the word "red": [CLS] a shiny [MASK] fire truck [SEP]. Then we gave the sentence to BERT and put the output to a softmax layer to get a proper probability value. The BERT's top 5 predictions for [MASK] are as follows:

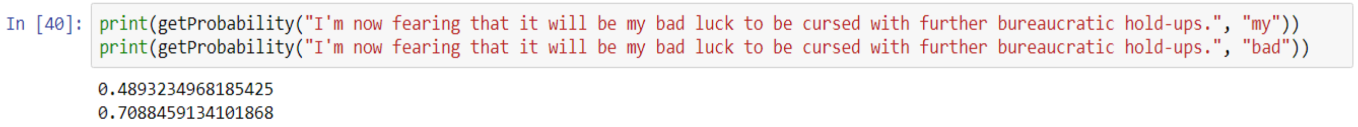
- Red: 0.41
- Black: 0.29
- New: 0.048
- Blue: 0.042
- Green: 0.038

In our case, we are only interested in the probability of the answer word that is red. In this case, the probability of "red" is quite low since even though it makes more sense for the masked word to be "red," it can also be any other adjective. However, if we perform this evaluation, then select the sentence with the highest probability, the output is usually very guessable. For example, running BERT on all the example sentences of "red" on wordhippo.com, we get the following output: ('A referee can show a yellow or _____ card to a player, substitute or substituted player.', 0.99). This is indeed an easily guessable sentence. However, not every word has this kind of good example sentences available; therefore, we implement a threshold such that our algorithm does not select the output of the BERT if the probability is lower than that threshold. One weakness of this method is that the answer word can only be exactly one word. It cannot be two words because BERT cannot interpret the two masked words when they are placed consequently. For example, if our answer is "my bad," even though it is possible to calculate the values of p(my|bad) and p(bad|my), it is impossible to find p(my, bad).



```
In [39]: getProbability2("[CLS] I'm now fearing that it will be [MASK] [MASK] luck to be cursed with further bureaucratic hold-ups. [SEP]'

Out[39]: 'my: 0.5327426791191101, bad: 0.029893090948462486'
```

Figure 1: The nonsensical output when BERT is asked to predict two words



```
In [40]: print(getProbability("I'm now fearing that it will be my bad luck to be cursed with further bureaucratic hold-ups.", "my"))
         print(getProbability("I'm now fearing that it will be my bad luck to be cursed with further bureaucratic hold-ups.", "bad"))

         0.4893234968185425
         0.7088459134101868
```

Figure 2: The probabilities p(my|bad) and p(bad|my).

Since there are plenty of stages, it is decided that when the answer is two words, it is not necessary to use the example sentence stage. Therefore, the clue generation from an example sentence only works when the input is strictly one word.

2. Definition and Knowledge Evaluation using Word2Vec

Even though it is simple to get definition sentences corresponding to a query from the Merriam-Webster API, there is still a need to select the best definition sentence possible for our clue. This requires evaluating those sentences and selecting the sentence with the highest score. For this purpose, a tool called Word2Vec is used. Word2Vec is a dictionary that has a corresponding vector form for each word. This makes it easy to process and compare the meaning of the words. A famous example of Word2Vec's would be "King - Man + Woman = Queen." Here it is seen that the output vectors do not represent the word's outer properties such as its letters or length but represent its meaning. Therefore, the Word2Vec data must be prepared using machine learning. However, since we do not use Word2Vec for a very specific purpose, but for general-purpose NLP, there was no need to train the data ourselves. Instead, we used the pre-trained Word2Vec dataset prepared by Google which was downloaded from **https://code.google.com/archive/p/word2vec/**. The reason for using Word2Vec instead of BERT in the definition sentences is because the definition of a word does not usually include the word itself. If the word isn't included in the sentence, the evaluation using BERT directly fails. However, definition sentences have a property that the example sentences do not have: all its words excluding the stop words (a, an, the, ...) have a correlation to the word it describes. This property is useful because the vector represented by the sum of all the main words, should be similar to the vector of the defined word. To give an example, assume that a definition of the answer "ugly" is "offensive to the sight: hideous." First, the stop words and punctuation are removed, and the remaining words are put in an array: ["offensive", "sight", "hideous"]. Then their vector sum is calculated:

$$\vec{v} = Word2Vec("offensive") + Word2Vec("sight") + Word2Vec("hideous")$$

Finally, it is necessary to find how similar v is to the vector form of "ugly." That similarity is calculated using a method called cosine similarity [2]. Cosine similarity is simply the cosine of the angle between two vectors. The calculation is straightforward: $\cos\theta = \frac{(\vec{u}\cdot\vec{v})}{||\vec{u}||||\vec{v}||}$. For this example, this similarity value is 0.55, which is quite high compared to the general case. This method's weakness is that it cannot differentiate between a grammatically correct and incorrect sentence; therefore, its input has to be from sources we trust to be grammatically correct such as a dictionary.

3. Google Knowledge Graph

Google Knowledge Graph inputs a query and output entities that have a common feature with that query. This is quite useful because the answer word can be "Cyrus," and we may not find any definition or example sentences about it. Therefore, there is a need for a system that gives us some knowledge about the answer word [3]. A part of the output of the GKG API when inputted "Cyrus" is as follows:

```
    {
      "@type": "EntitySearchResult",
      "result": {
        "@type": [
          "Thing",
          "Person"
        ],
        "description": "American singer-songwriter",
        "detailedDescription": {
          "license":
"https://en.wikipedia.org/wiki/Wikipedia:Text_of_Creative_Commons_Attribution-
ShareAlike_3.0_Unported_License",
          "url": "https://en.wikipedia.org/wiki/Miley_Cyrus",
          "articleBody": "Miley Ray Hemsworth is an American singer, songwriter, actress, and
philanthropist. Her music has spanned a range of styles, including pop, country pop, and hip
hop. "
        },
        "url": "http://www.mileycyrus.com/",
        "@id": "kg:/m/0bdxs5",
        "image": {
          "contentUrl": "https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRaKHrcDl-
V6JDgcQc-symj4PbE-H3a5kcpUPSS6hgdMqVXzR6K",
          "url": "https://commons.wikimedia.org/wiki/File:170526-N-EO381-
052_Miley_Cyrus_on_Today_show.jpg"
        },
        "name": "Miley Cyrus"
      },
      "resultScore": 2170.700927734375
    },
```

The information given here is quite useful and can easily be used in clue generation. For the evaluation of the possible clues, Word2Vec is used. However, there may be a word that exists in GKG but not in the Word2Vec vocabulary. In that case, GKG stage returns -1. If all clue generation stages fail and the GKG step had failed by -1, the longest GKG sentence is selected as the new clue.

4. Datamuse API

The Datamuse API is an easy to use word relation search engine. In our algorithm, this API is used to find rhyming words. This is a quite simple process. The response of the website https://api.datamuse.com/words?rel_rhy=[word] directly gives the rhyming words with the inputted word with their corresponding score. A sample output would be like the following:

```
 [{"word":"fretful","score":398,"numSyllables":2},
{"word":"regretful","score":303,"numSyllables":3},
{"word":"threatful","score":129,"numSyllables":2}]
```

There is a numerical evaluation for every word, and they are already sorted from the largest score to lowest. In our case, the first entry is good enough since the algorithm just requires a rhyming word.

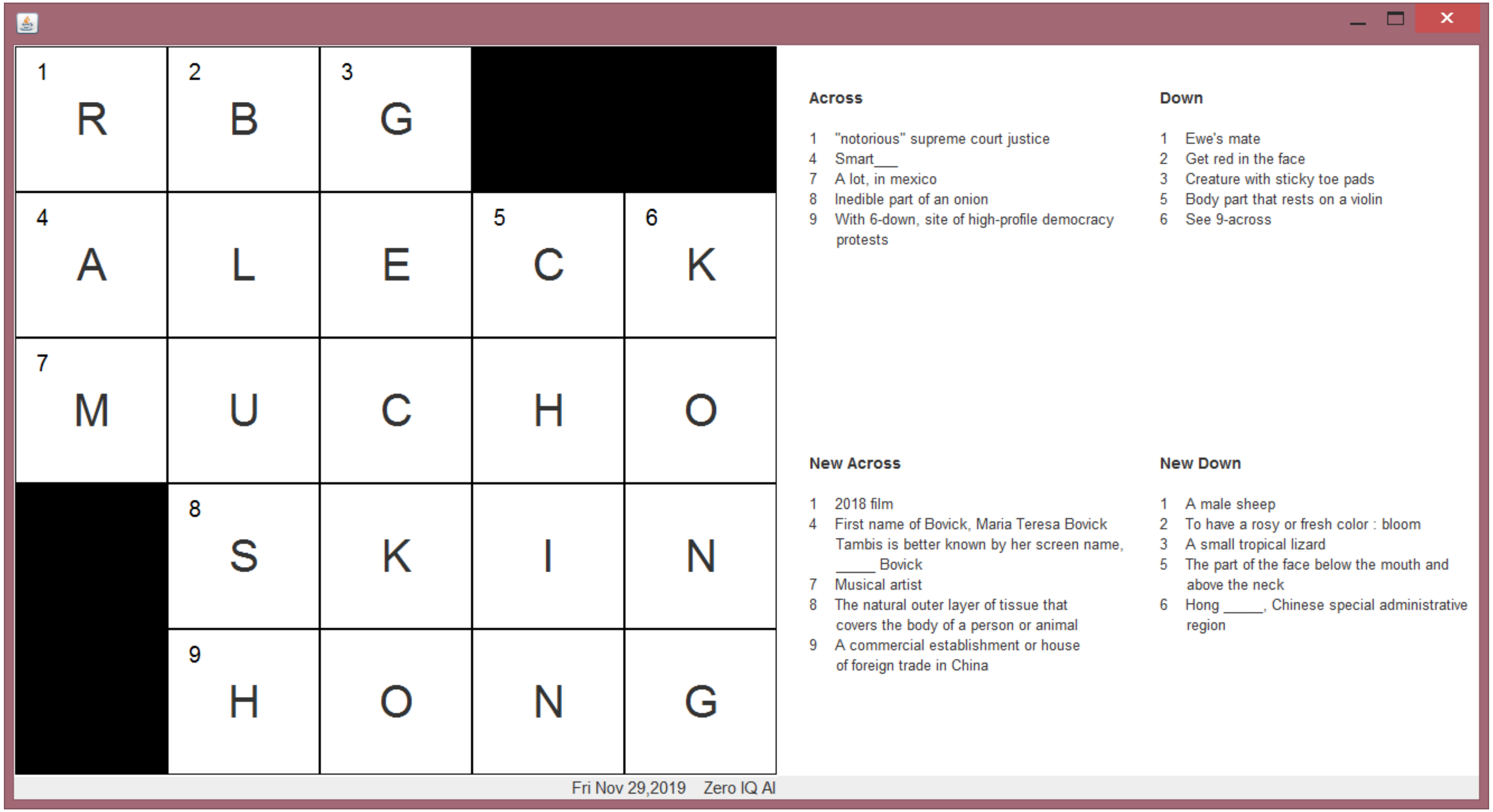
Aside from rhyming words, this API is used to split words. For example, the answer "mybad" is not one word but two. In order to split this, the suggestion feature of the Datamuse is used. The suggestion feature directly splits the word assuming it was miswritten; therefore, it is convenient for our algorithm.

5. Detailed Algorithm Flow

```
1.  Start
2.  Scrape Merriam-Webster for Definition and Example Sentences using its API
    2.1.  Filter out the sentences whose edit distance to the old clue is less than a threshold
    2.2.  Categorize data into 3 parts: Definitions, Examples, One-word Examples
    2.3.  If at least one definition is found
          2.3.1.  Let v be the vector representation of the answer word
          2.3.2.  For each definition sentence
                  2.3.2.1.  Let u be the sum of the vector representations of all the words in
                            the sentence
                  2.3.2.2.  Find the cosine similarity (cosine of the angle) between v and u
          2.3.3.  Choose the sentence with the highest similarity value as the new clue
          2.3.4.  End
```

```
3.      Scrape example sentences from wordhippo.com and concatenate with the example
        sentences from Merriam-Webster
          3.1.1.    Filter out the sentences that includes the answer word more than once
          3.1.2.    If at least one example sentence available
              3.1.2.1.    For each sentence, use BERT to get a probability of the answer
                          word belonging to that sentence and select the one with the highest
                          probability
              3.1.2.2.    If the probability of greater than some threshold
                  3.1.2.2.1.    Choose that sentence as the new clue
                  3.1.2.2.2.    End.
4.      Search for a sense GKG
        4.1.    If answer word exists in imported Word2Vec vocabulary
            4.1.1.    Let v be the vector representation of the answer word
            4.1.2.    For each acquired sentence
                4.1.2.1.    Let u be the sum of the vector representations of all the words in
                            the sentence
                4.1.2.2.    Find the cosine similarity between v and u
            4.1.3.    Select the sentence with highest similarity value
            4.1.4.    If the similarity is greater than some threshold
                4.1.4.1.    Choose that sentence as the new clue
                4.1.4.2.    End.
        4.2.    Else (Answer not in vocabulary) keep the longest sentence for later
5.      If there exist one-word examples
        5.1.    Select the answer as a prefix or suffix of the one-word example (Example: "Prefix
                with -lexia")
        5.2.    End
6.      If a sentence available from step 4.2.
        6.1.    Select that as the new clue
7.      Start from the beginning with the search word as the stem of the answer
8.      Check if the answer is actually two words combined. If that is the case, start from the
        beginning with the new search word as the split form of the answer.
9.      Check Datamuse API to see whether there are rhyming words with the answer. If there
        exists at least one rhyming word
        9.1.    Select the new clue as "It rhymes with …"
        9.2.    End.
10.     The new clue couldn't be generated.
11.     End.
```

Results



1 R	2 B	3 G	■	■
4 A	L	E	5 C	6 K
7 M	U	C	H	O
■	8 S	K	I	N
■	9 H	O	N	G

Across

1 "notorious" supreme court justice
4 Smart___
7 A lot, in mexico
8 Inedible part of an onion
9 With 6-down, site of high-profile democracy protests

Down

1 Ewe's mate
2 Get red in the face
3 Creature with sticky toe pads
5 Body part that rests on a violin
6 See 9-across

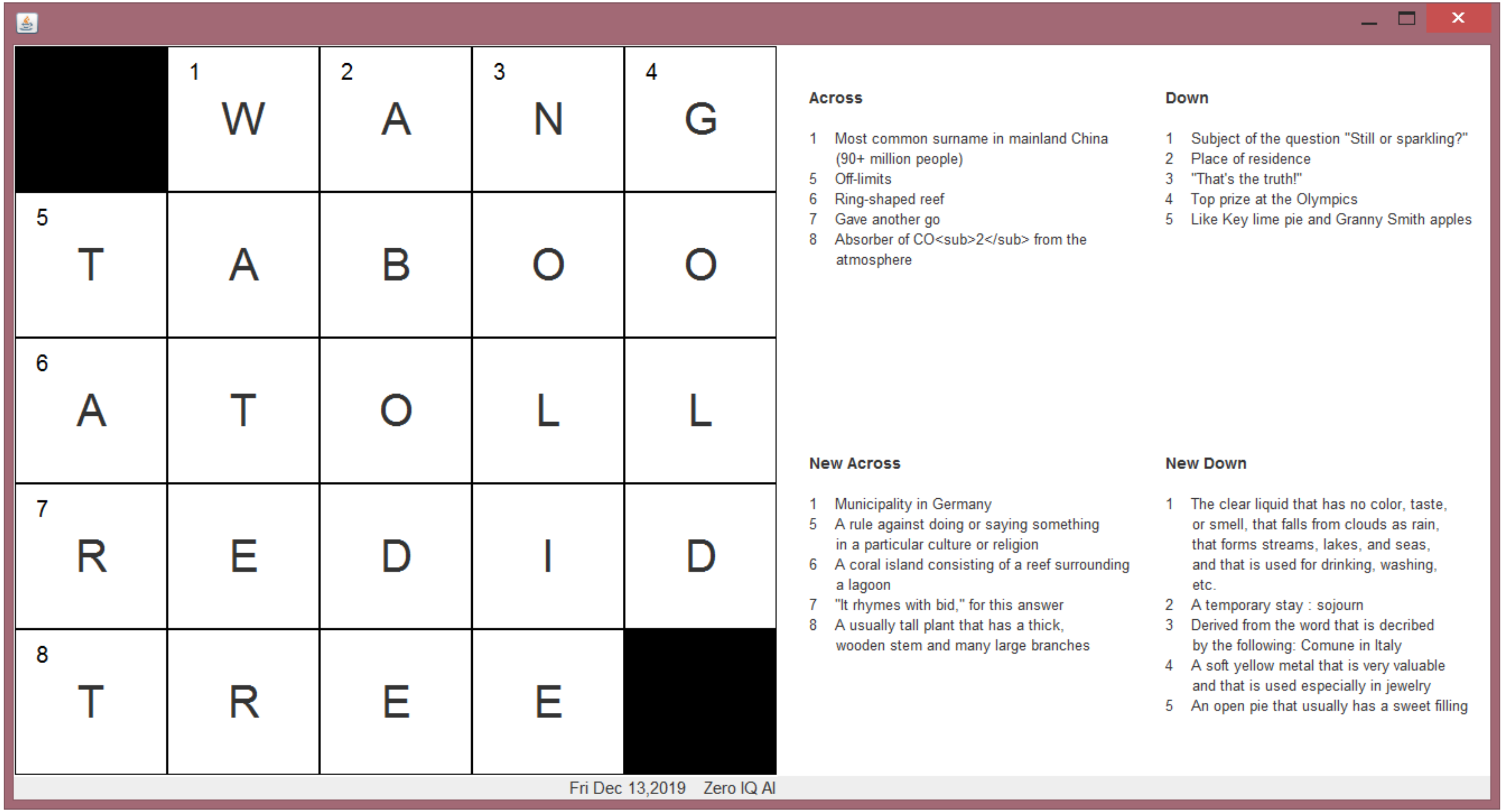
New Across

1 2018 film
4 First name of Bovick, Maria Teresa Bovick Tambis is better known by her screen name, _____ Bovick
7 Musical artist
8 The natural outer layer of tissue that covers the body of a person or animal
9 A commercial establishment or house of foreign trade in China

New Down

1 A male sheep
2 To have a rosy or fresh color : bloom
3 A small tropical lizard
5 The part of the face below the mouth and above the neck
6 Hong _____, Chinese special administrative region

Fri Nov 29,2019 Zero IQ AI



■	1 W	2 A	3 N	4 G
5 T	A	B	O	O
6 A	T	O	L	L
7 R	E	D	I	D
8 T	R	E	E	■

Across

1 Most common surname in mainland China (90+ million people)
5 Off-limits
6 Ring-shaped reef
7 Gave another go
8 Absorber of CO₂ from the atmosphere

Down

1 Subject of the question "Still or sparkling?"
2 Place of residence
3 "That's the truth!"
4 Top prize at the Olympics
5 Like Key lime pie and Granny Smith apples

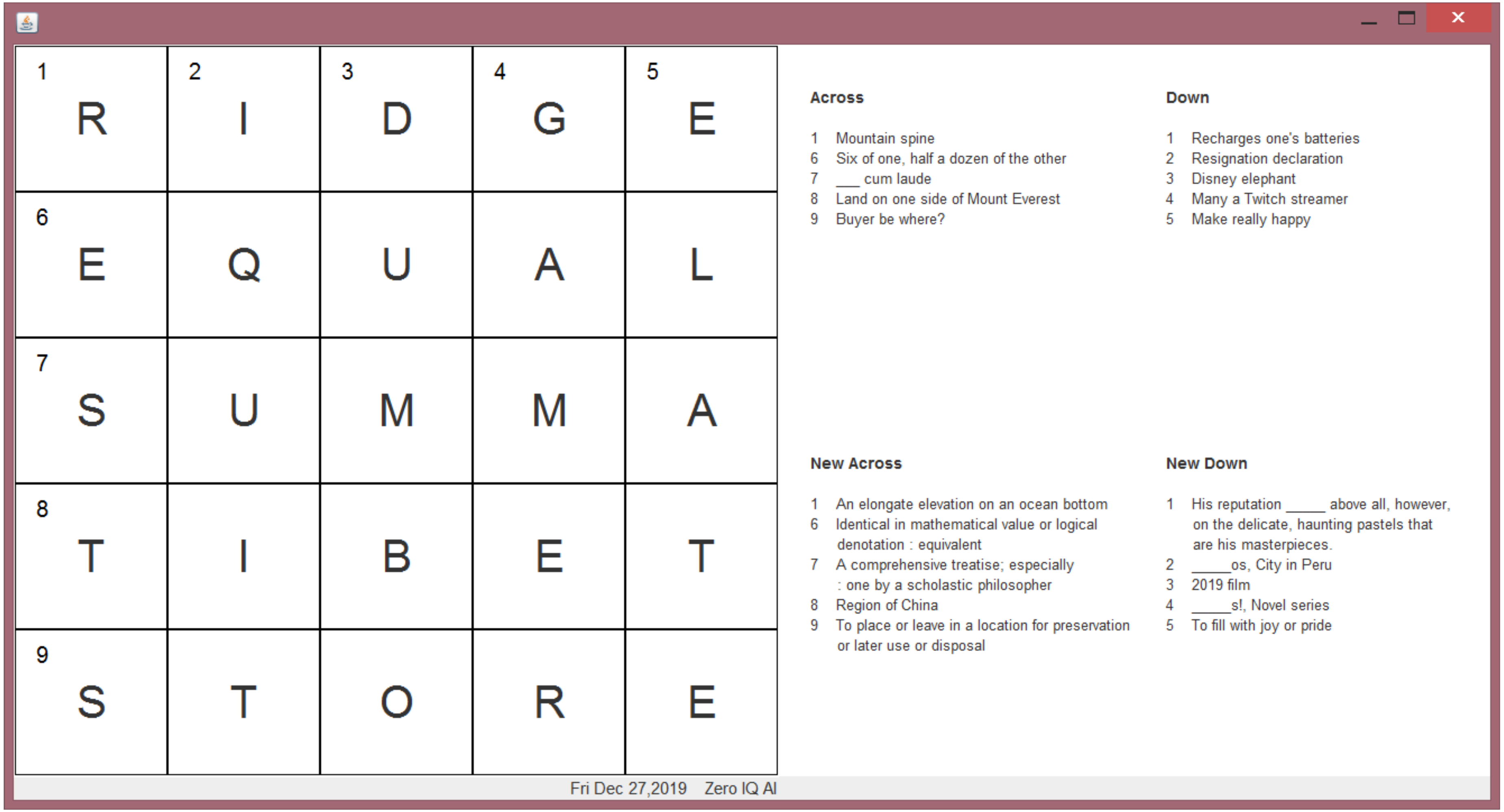
New Across

1 Municipality in Germany
5 A rule against doing or saying something in a particular culture or religion
6 A coral island consisting of a reef surrounding a lagoon
7 "It rhymes with bid," for this answer
8 A usually tall plant that has a thick, wooden stem and many large branches

New Down

1 The clear liquid that has no color, taste, or smell, that falls from clouds as rain, that forms streams, lakes, and seas, and that is used for drinking, washing, etc.
2 A temporary stay : sojourn
3 Derived from the word that is decribed by the following: Comune in Italy
4 A soft yellow metal that is very valuable and that is used especially in jewelry
5 An open pie that usually has a sweet filling

Fri Dec 13,2019 Zero IQ AI



1 R	2 I	3 D	4 G	5 E
6 E	Q	U	A	L
7 S	U	M	M	A
8 T	I	B	E	T
9 S	T	O	R	E

Across

1 Mountain spine
6 Six of one, half a dozen of the other
7 ___ cum laude
8 Land on one side of Mount Everest
9 Buyer be where?

Down

1 Recharges one's batteries
2 Resignation declaration
3 Disney elephant
4 Many a Twitch streamer
5 Make really happy

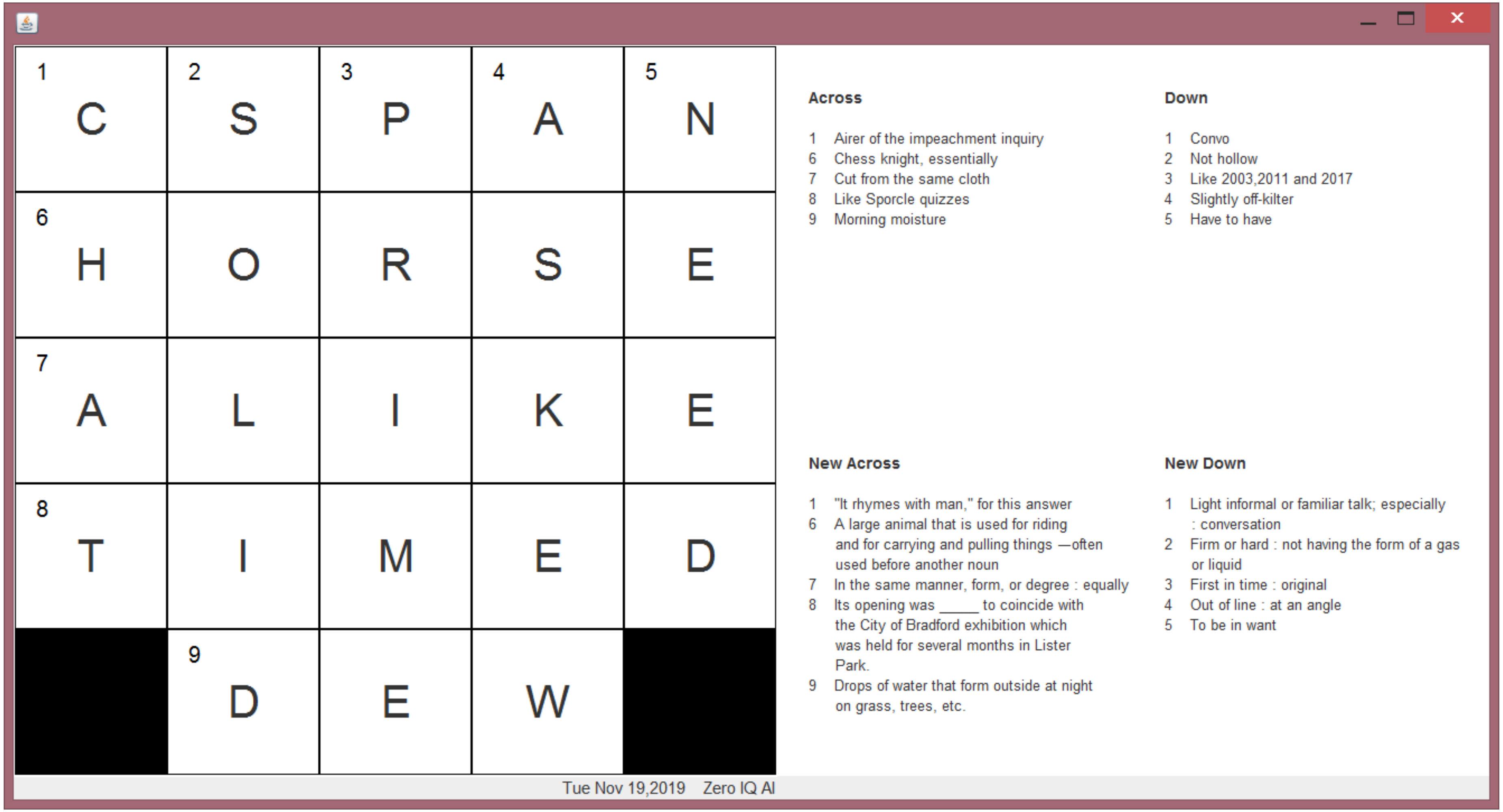
New Across

1 An elongate elevation on an ocean bottom
6 Identical in mathematical value or logical denotation : equivalent
7 A comprehensive treatise; especially : one by a scholastic philosopher
8 Region of China
9 To place or leave in a location for preservation or later use or disposal

New Down

1 His reputation _____ above all, however, on the delicate, haunting pastels that are his masterpieces.
2 _____os, City in Peru
3 2019 film
4 _____s!, Novel series
5 To fill with joy or pride

Fri Dec 27,2019 Zero IQ AI



1 C	2 S	3 P	4 A	5 N
6 H	O	R	S	E
7 A	L	I	K	E
8 T	I	M	E	D
■	9 D	E	W	■

Across

1 Airer of the impeachment inquiry
6 Chess knight, essentially
7 Cut from the same cloth
8 Like Sporcle quizzes
9 Morning moisture

Down

1 Convo
2 Not hollow
3 Like 2003,2011 and 2017
4 Slightly off-kilter
5 Have to have

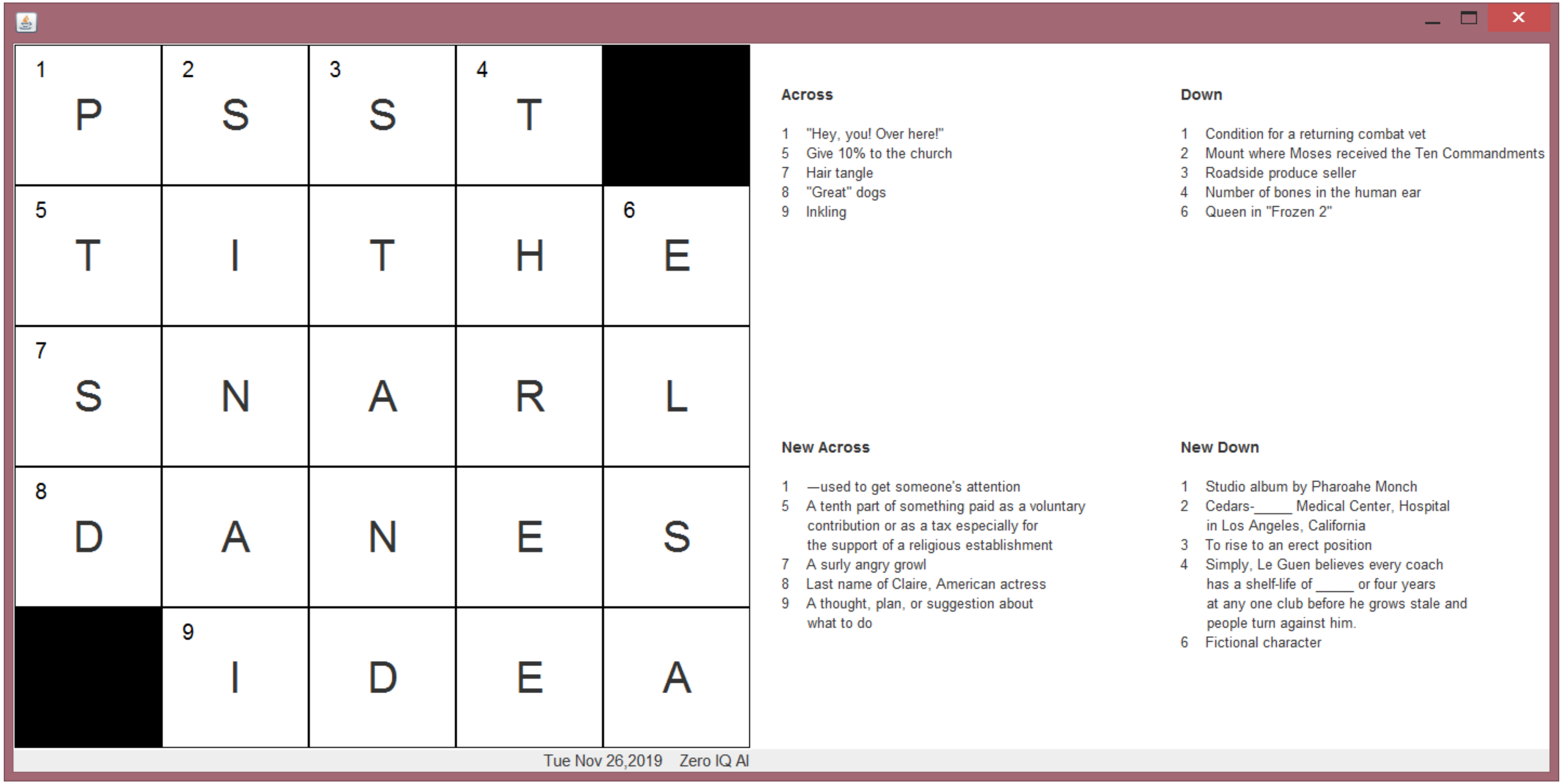
New Across

1 "It rhymes with man," for this answer
6 A large animal that is used for riding and for carrying and pulling things —often used before another noun
7 In the same manner, form, or degree : equally
8 Its opening was _____ to coincide with the City of Bradford exhibition which was held for several months in Lister Park.
9 Drops of water that form outside at night on grass, trees, etc.

New Down

1 Light informal or familiar talk; especially : conversation
2 Firm or hard : not having the form of a gas or liquid
3 First in time : original
4 Out of line : at an angle
5 To be in want

Tue Nov 19,2019 Zero IQ AI



1 P	2 S	3 S	4 T	
5 T	I	T	H	6 E
7 S	N	A	R	L
8 D	A	N	E	S
	9 I	D	E	A

Across

1 "Hey, you! Over here!"
5 Give 10% to the church
7 Hair tangle
8 "Great" dogs
9 Inkling

Down

1 Condition for a returning combat vet
2 Mount where Moses received the Ten Commandments
3 Roadside produce seller
4 Number of bones in the human ear
6 Queen in "Frozen 2"

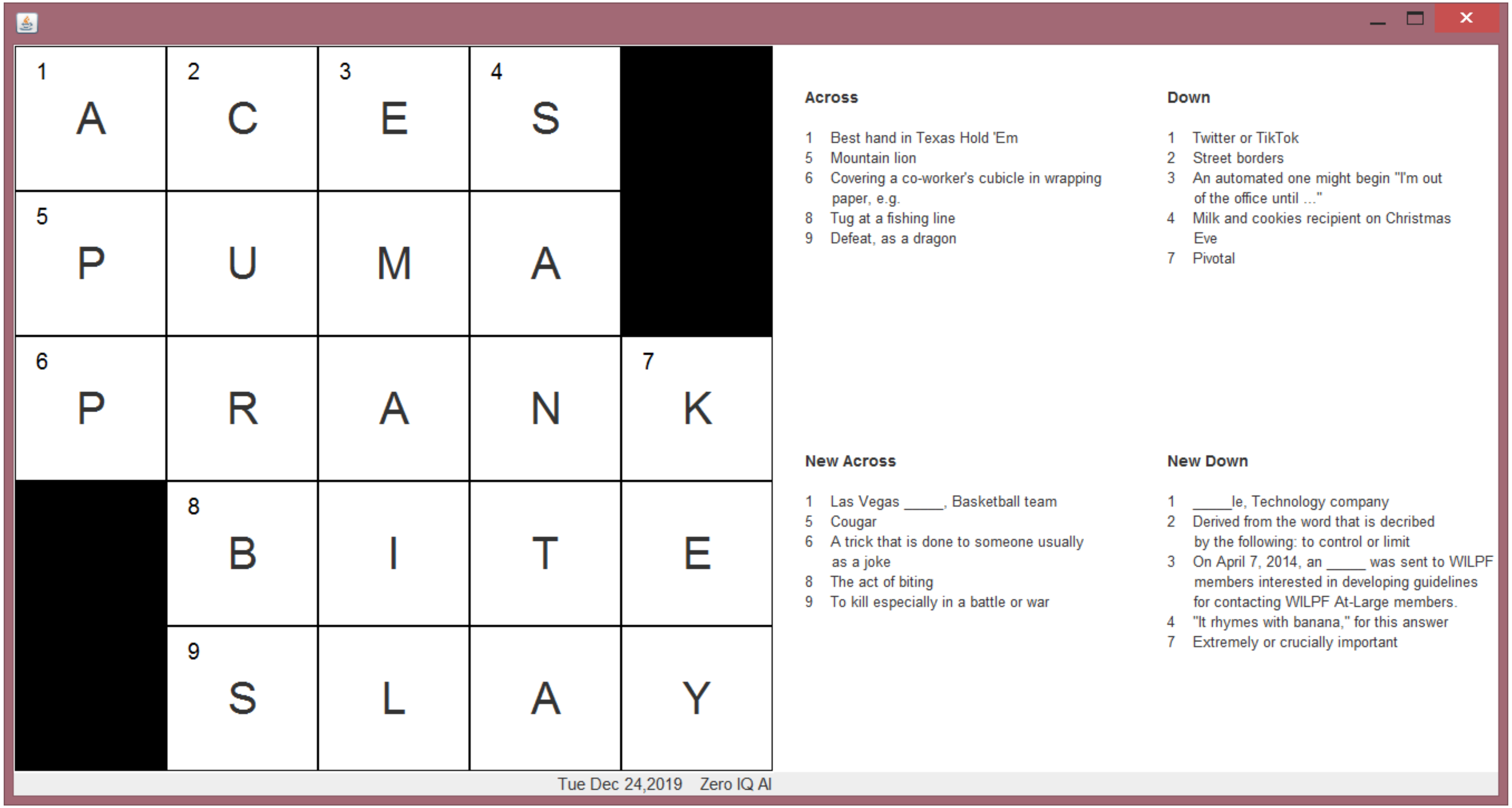
New Across

1 —used to get someone's attention
5 A tenth part of something paid as a voluntary contribution or as a tax especially for the support of a religious establishment
7 A surly angry growl
8 Last name of Claire, American actress
9 A thought, plan, or suggestion about what to do

New Down

1 Studio album by Pharoahe Monch
2 Cedars-_____ Medical Center, Hospital in Los Angeles, California
3 To rise to an erect position
4 Simply, Le Guen believes every coach has a shelf-life of _____ or four years at any one club before he grows stale and people turn against him.
6 Fictional character

Tue Nov 26,2019 Zero IQ AI



1 A	2 C	3 E	4 S	
5 P	U	M	A	
6 P	R	A	N	7 K
	8 B	I	T	E
	9 S	L	A	Y

Across

1 Best hand in Texas Hold 'Em
5 Mountain lion
6 Covering a co-worker's cubicle in wrapping paper, e.g.
8 Tug at a fishing line
9 Defeat, as a dragon

Down

1 Twitter or TikTok
2 Street borders
3 An automated one might begin "I'm out of the office until ..."
4 Milk and cookies recipient on Christmas Eve
7 Pivotal

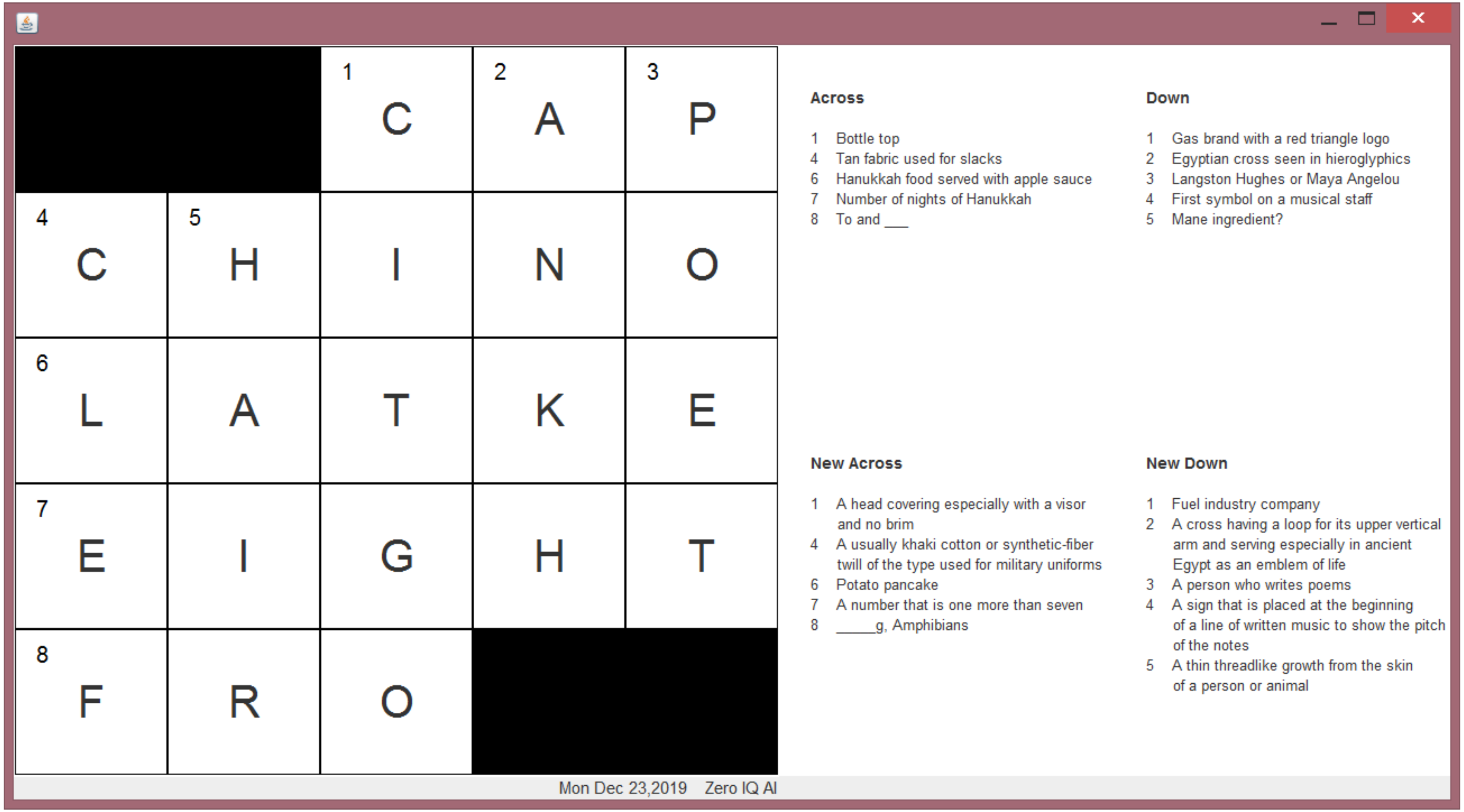
New Across

1 Las Vegas _____, Basketball team
5 Cougar
6 A trick that is done to someone usually as a joke
8 The act of biting
9 To kill especially in a battle or war

New Down

1 _____le, Technology company
2 Derived from the word that is decribed by the following: to control or limit
3 On April 7, 2014, an _____ was sent to WILPF members interested in developing guidelines for contacting WILPF At-Large members.
4 "It rhymes with banana," for this answer
7 Extremely or crucially important

Tue Dec 24,2019 Zero IQ AI



■	■	1 C	2 A	3 P
4 C	5 H	I	N	O
6 L	A	T	K	E
7 E	I	G	H	T
8 F	R	O	■	■

Across

1 Bottle top
4 Tan fabric used for slacks
6 Hanukkah food served with apple sauce
7 Number of nights of Hanukkah
8 To and ___

Down

1 Gas brand with a red triangle logo
2 Egyptian cross seen in hieroglyphics
3 Langston Hughes or Maya Angelou
4 First symbol on a musical staff
5 Mane ingredient?

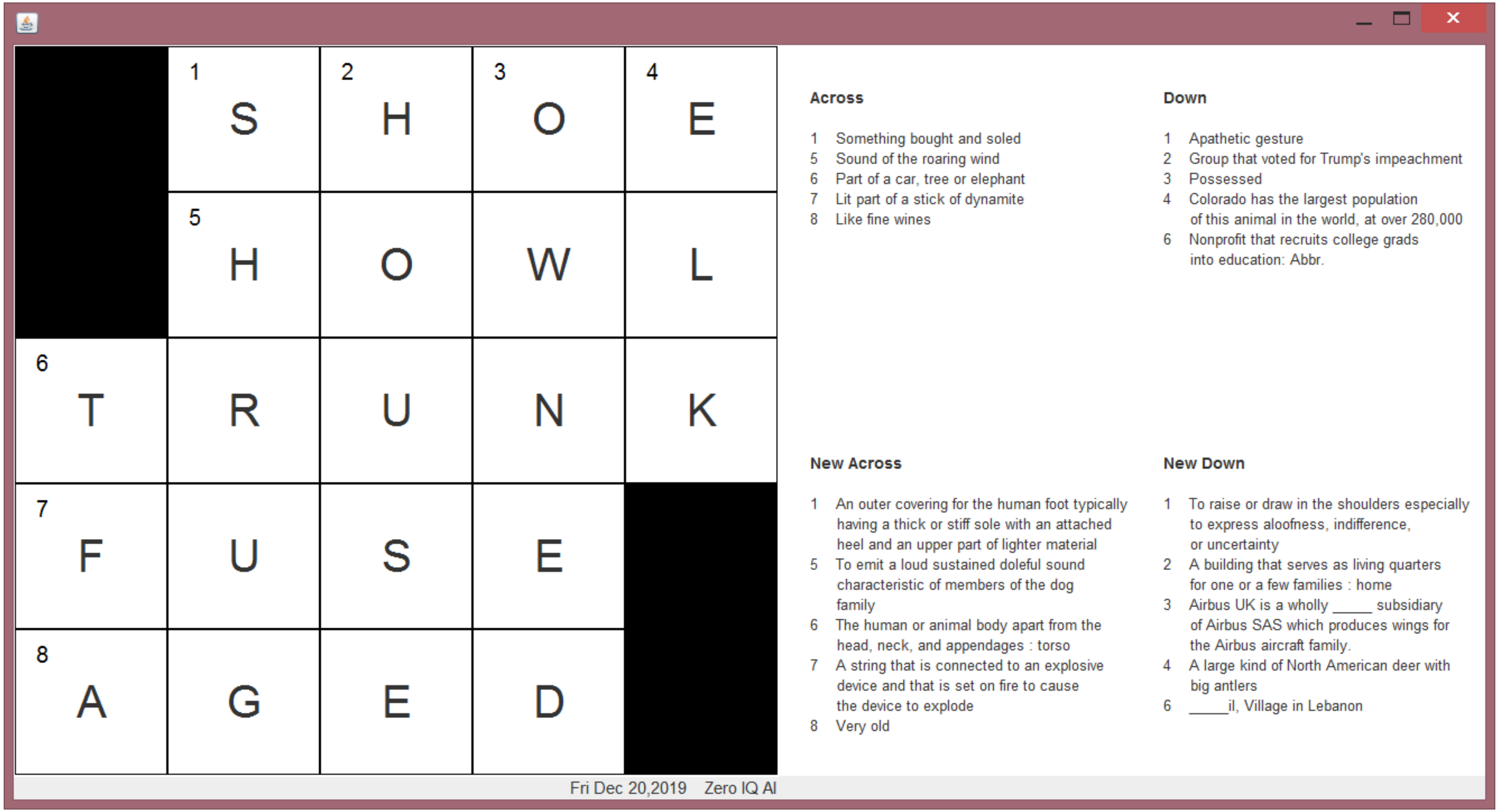
New Across

1 A head covering especially with a visor and no brim
4 A usually khaki cotton or synthetic-fiber twill of the type used for military uniforms
6 Potato pancake
7 A number that is one more than seven
8 _____g, Amphibians

New Down

1 Fuel industry company
2 A cross having a loop for its upper vertical arm and serving especially in ancient Egypt as an emblem of life
3 A person who writes poems
4 A sign that is placed at the beginning of a line of written music to show the pitch of the notes
5 A thin threadlike growth from the skin of a person or animal

Mon Dec 23,2019 Zero IQ AI



	1 S	2 H	3 O	4 E
	5 H	O	W	L
6 T	R	U	N	K
7 F	U	S	E	
8 A	G	E	D	

Across

1 Something bought and soled
5 Sound of the roaring wind
6 Part of a car, tree or elephant
7 Lit part of a stick of dynamite
8 Like fine wines

Down

1 Apathetic gesture
2 Group that voted for Trump's impeachment
3 Possessed
4 Colorado has the largest population of this animal in the world, at over 280,000
6 Nonprofit that recruits college grads into education: Abbr.

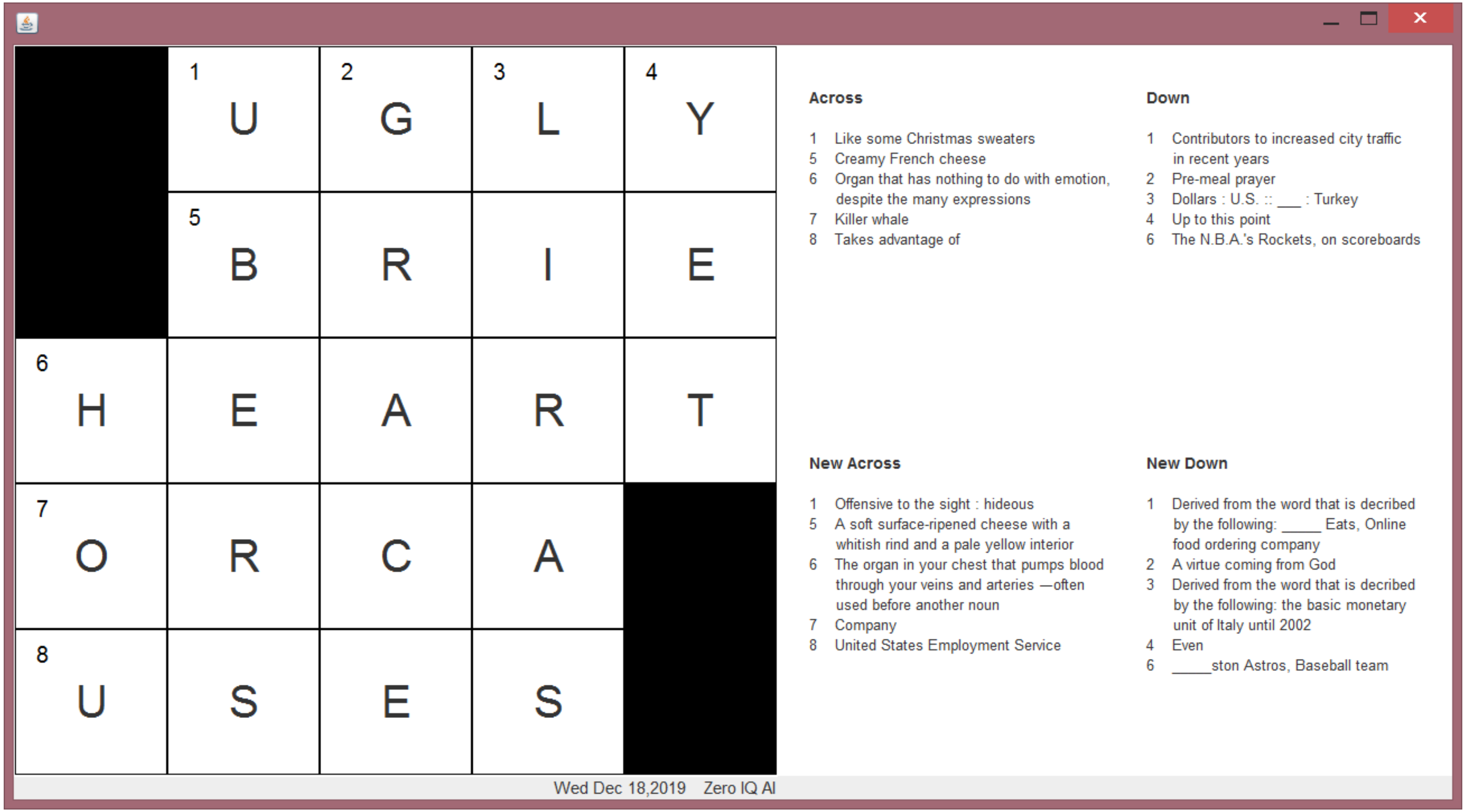
New Across

1 An outer covering for the human foot typically having a thick or stiff sole with an attached heel and an upper part of lighter material
5 To emit a loud sustained doleful sound characteristic of members of the dog family
6 The human or animal body apart from the head, neck, and appendages : torso
7 A string that is connected to an explosive device and that is set on fire to cause the device to explode
8 Very old

New Down

1 To raise or draw in the shoulders especially to express aloofness, indifference, or uncertainty
2 A building that serves as living quarters for one or a few families : home
3 Airbus UK is a wholly _____ subsidiary of Airbus SAS which produces wings for the Airbus aircraft family.
4 A large kind of North American deer with big antlers
6 _____il, Village in Lebanon

Fri Dec 20,2019 Zero IQ AI



	1 U	2 G	3 L	4 Y
	5 B	R	I	E
6 H	E	A	R	T
7 O	R	C	A	
8 U	S	E	S	

Across

1 Like some Christmas sweaters
5 Creamy French cheese
6 Organ that has nothing to do with emotion, despite the many expressions
7 Killer whale
8 Takes advantage of

Down

1 Contributors to increased city traffic in recent years
2 Pre-meal prayer
3 Dollars : U.S. :: ___ : Turkey
4 Up to this point
6 The N.B.A.'s Rockets, on scoreboards

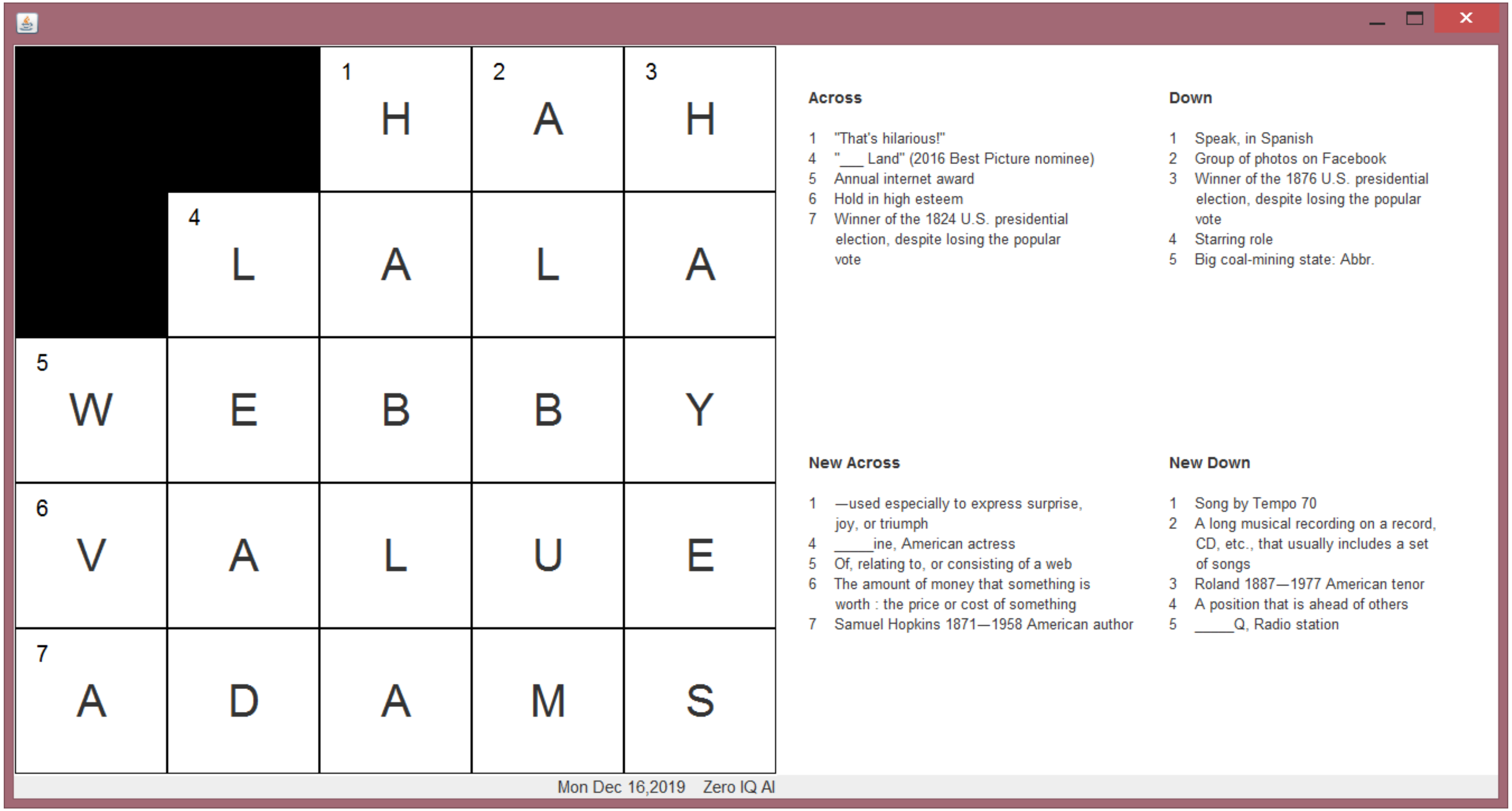
New Across

1 Offensive to the sight : hideous
5 A soft surface-ripened cheese with a whitish rind and a pale yellow interior
6 The organ in your chest that pumps blood through your veins and arteries —often used before another noun
7 Company
8 United States Employment Service

New Down

1 Derived from the word that is decribed by the following: _____ Eats, Online food ordering company
2 A virtue coming from God
3 Derived from the word that is decribed by the following: the basic monetary unit of Italy until 2002
4 Even
6 _____ston Astros, Baseball team

Wed Dec 18,2019 Zero IQ AI



		1 H	2 A	3 H
	4 L	A	L	A
5 W	E	B	B	Y
6 V	A	L	U	E
7 A	D	A	M	S

Across

1 "That's hilarious!"
4 "___ Land" (2016 Best Picture nominee)
5 Annual internet award
6 Hold in high esteem
7 Winner of the 1824 U.S. presidential election, despite losing the popular vote

Down

1 Speak, in Spanish
2 Group of photos on Facebook
3 Winner of the 1876 U.S. presidential election, despite losing the popular vote
4 Starring role
5 Big coal-mining state: Abbr.

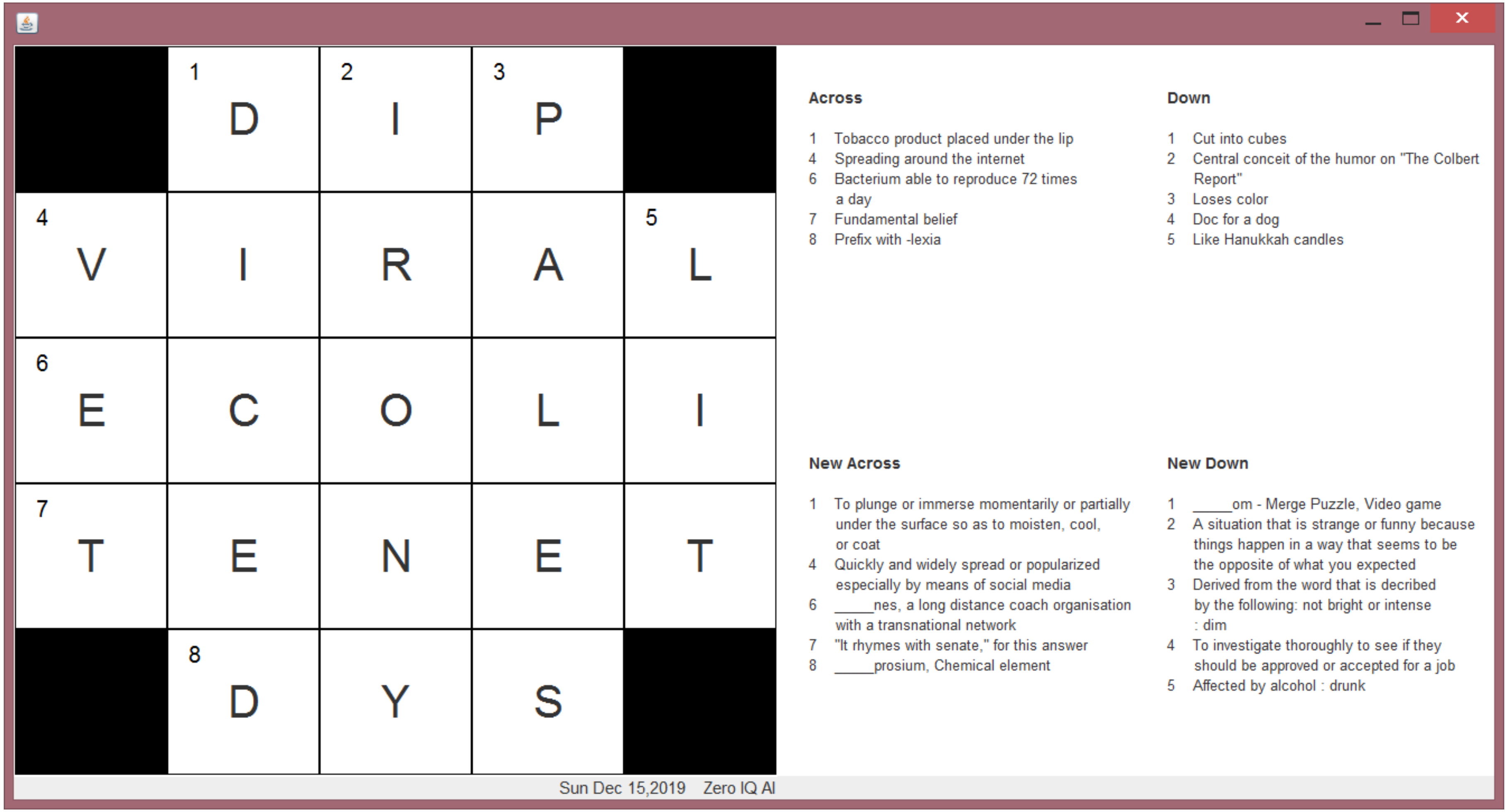
New Across

1 —used especially to express surprise, joy, or triumph
4 _____ine, American actress
5 Of, relating to, or consisting of a web
6 The amount of money that something is worth : the price or cost of something
7 Samuel Hopkins 1871—1958 American author

New Down

1 Song by Tempo 70
2 A long musical recording on a record, CD, etc., that usually includes a set of songs
3 Roland 1887—1977 American tenor
4 A position that is ahead of others
5 _____Q, Radio station

Mon Dec 16,2019 Zero IQ AI



	1 D	2 I	3 P	
4 V	I	R	A	5 L
6 E	C	O	L	I
7 T	E	N	E	T
	8 D	Y	S	

Across

1 Tobacco product placed under the lip
4 Spreading around the internet
6 Bacterium able to reproduce 72 times a day
7 Fundamental belief
8 Prefix with -lexia

Down

1 Cut into cubes
2 Central conceit of the humor on "The Colbert Report"
3 Loses color
4 Doc for a dog
5 Like Hanukkah candles

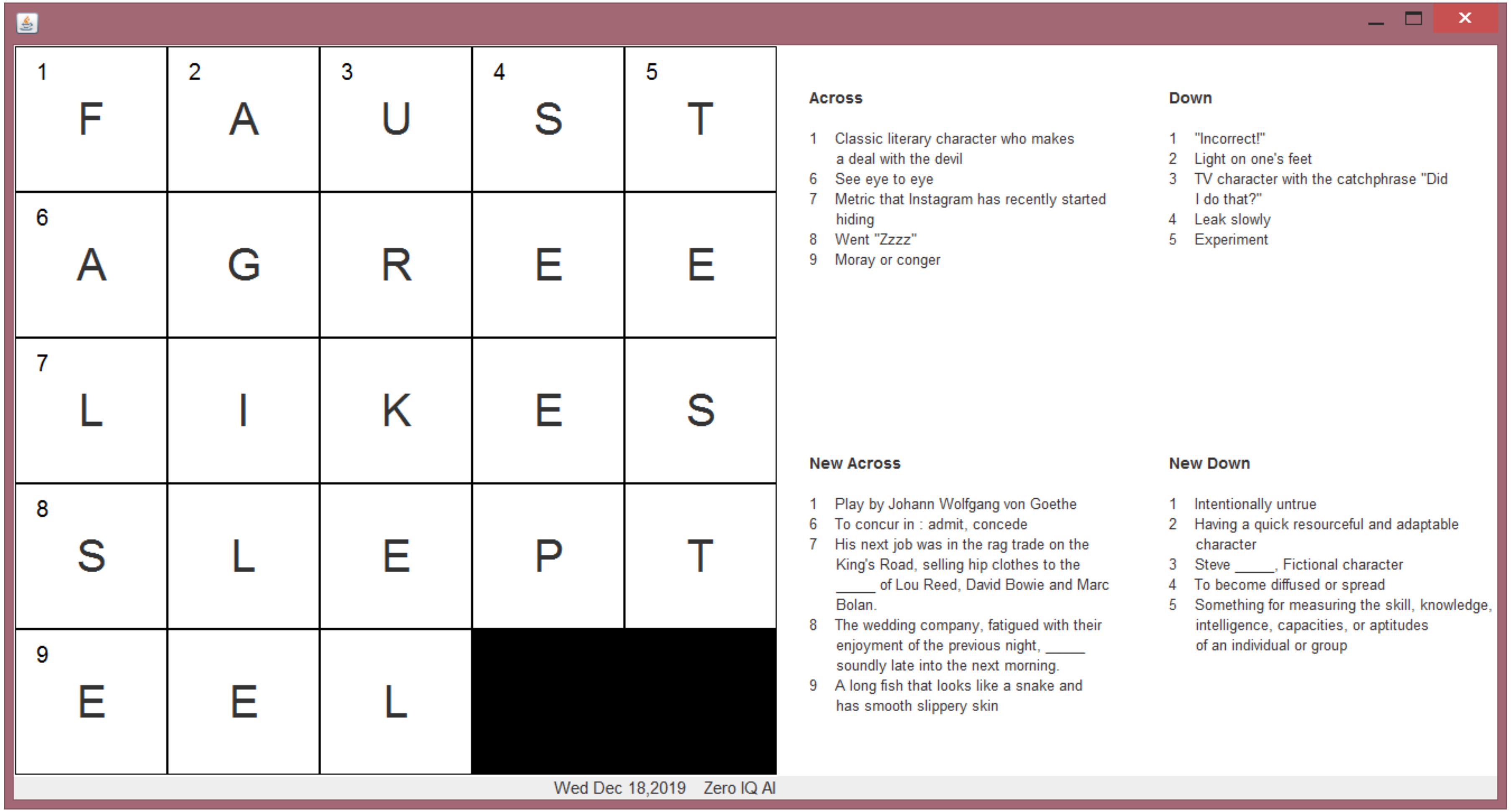
New Across

1 To plunge or immerse momentarily or partially under the surface so as to moisten, cool, or coat
4 Quickly and widely spread or popularized especially by means of social media
6 _____nes, a long distance coach organisation with a transnational network
7 "It rhymes with senate," for this answer
8 _____prosium, Chemical element

New Down

1 _____om - Merge Puzzle, Video game
2 A situation that is strange or funny because things happen in a way that seems to be the opposite of what you expected
3 Derived from the word that is decribed by the following: not bright or intense : dim
4 To investigate thoroughly to see if they should be approved or accepted for a job
5 Affected by alcohol : drunk

Sun Dec 15,2019 Zero IQ AI



1 F	2 A	3 U	4 S	5 T
6 A	G	R	E	E
7 L	I	K	E	S
8 S	L	E	P	T
9 E	E	L	■	■

Across

1 Classic literary character who makes a deal with the devil
6 See eye to eye
7 Metric that Instagram has recently started hiding
8 Went "Zzzz"
9 Moray or conger

Down

1 "Incorrect!"
2 Light on one's feet
3 TV character with the catchphrase "Did I do that?"
4 Leak slowly
5 Experiment

New Across

1 Play by Johann Wolfgang von Goethe
6 To concur in : admit, concede
7 His next job was in the rag trade on the King's Road, selling hip clothes to the _____ of Lou Reed, David Bowie and Marc Bolan.
8 The wedding company, fatigued with their enjoyment of the previous night, _____ soundly late into the next morning.
9 A long fish that looks like a snake and has smooth slippery skin

New Down

1 Intentionally untrue
2 Having a quick resourceful and adaptable character
3 Steve _____, Fictional character
4 To become diffused or spread
5 Something for measuring the skill, knowledge, intelligence, capacities, or aptitudes of an individual or group

Wed Dec 18,2019 Zero IQ AI

This project reports work done in partial fulfillment of the requirements for CS 461 -- Artificial Intelligence. The software is, to a large extent, original (with borrowed code clearly identified) and was written solely by members of the project group ZERO IQ AI.

Word Count: 2480 words

References

[1] Vig, J., 2019, “Bertviz: A Tool for Visualizing Multi-Head Self-Attention in the Bert Model,” pp. 1-6 in: ICLR 2019 Debugging Machine Learning Models Workshop, New Orleans

[2] Jatnika, D., Bijaksana, M. & Suryani A., 2019, “Word2Vec Model Analysis for Semantic Similarities in English Words,” pp. 160-167 in: 4th International Conference on Computer Science and Computational Intelligence, Bandung

[3] Suchanek, F. & Weikum, G., 2013, “Knowledge harvesting in the big-data era,” pp. 933-938 in: Proceedings of the ACM SIGMOD International Conference on Management of Data, New York

Appendices

Appendix A: NY Times Scraper written in Python

```
#!/usr/bin/env python
# coding: utf-8
"""
Created on Tue Oct 1 18:45:34 2019

@author: Ege Aydin
"""

from bs4 import BeautifulSoup
from selenium import webdriver
from selenium.webdriver.firefox.options import Options
import re
import numpy as np
import html

print("Loading...")
timeout = 10
puzzle = np.array([''] * 25)
clues = []
clue_numbers = []
puzzle_url = "https://www.nytimes.com/crosswords/game/mini"
#"https://www.nytimes.com/crosswords/game/special/tricky-clues-mini" #

options = Options()
options.headless = True
driver = webdriver.Firefox(options=options)

driver.get(puzzle_url)
driver.implicitly_wait(timeout)

driver.find_element_by_xpath("//*[text()='OK']").click()
driver.find_element_by_xpath("//*[text()='reveal']").click()
driver.find_elements_by_xpath("//*[text()='Puzzle']")[1].click()
driver.find_element_by_xpath("//*[text()='Reveal']").click()
driver.find_element_by_class_name("ModalBody-closeX--2Fmp7").click()

page_source = driver.page_source
driver.close()

soup = BeautifulSoup(page_source, 'html.parser')
for element in soup.findAll(class_="Cell-block--1oNaD"):
    puzzle[int(element.get("id")[8:])] = '#'

for element in soup.findAll(class_="Clue-text--3lZl7"):
    clues.append(html.unescape(re.search("<span class=\"Clue-text--3lZl7\">(.*?)</span>",\
  str(element)).group()[31:-7]))

for element in soup.findAll(class_="Clue-label--2IdMY"):
    clue_numbers.append(int(re.search(">(.*?)<", str(element)).group()[1:-1]))

counter = 0
```

```
digit_places = {}

#print(puzzle, clues, clue_numbers)
#print(re.findall("(>[A-Z1-9]<)", str(soup.findAll("text"))))

letters = re.findall("(>[A-Z1-9]<)", str(soup.findAll("text")))

store = 0
old_letter = ''
for e in range(len(letters)-1, -1, -1):
    if letters[e][1].isalpha():
        if letters[e][1] == old_letter:
            store += 1
        else:
            store = 0
            old_letter = letters[e][1]
        if store == 2:
            del letters[e]
            del letters[e+1]
            old_letter = ''
            store = 0
    else:
        old_letter = ''
        store = 0


for e in letters:
    print(e)
    while puzzle[counter] == '#':
        counter += 1
    if e[1].isdigit():
        digit_places[int(e[1])] = (counter // 5, counter % 5)
    else:
        puzzle[counter] = html.unescape(e[1])
        counter += 1
# put things to words
puzzle = puzzle.reshape((5, 5))
wordlist = []

rotated_puzzle = np.rot90(puzzle)

for e in clue_numbers[:5]:
    wordlist.append(''.join(puzzle[digit_places[e][0]]))

for e in clue_numbers[5:]:
    wordlist.append(''.join(rotated_puzzle[4-digit_places[e][1]]))


str_digit_places = str(digit_places).replace(" ", "").replace("),", "); ")

print(wordlist)
print(clues)
print(clue_numbers)
print(str_digit_places)
print(puzzle)

clues = [clue.encode(encoding = 'ascii', errors = 'xmlcharrefreplace').decode() for\
  clue in clues]

np.savetxt("aiFood.txt", [clue_numbers, wordlist, clues, str_digit_places,
puzzle.reshape(25)],\
    delimiter=",", fmt="%s")
```

Appendix B: GUI written in Java

Launcher.java

```
import java.io.File;
import java.io.FileNotFoundException;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.Scanner;
/**
 * @author Denizhan Soydas
 * @author Onur Kirmizi
 * @author Ege Aydin
 * This class is the Launcher of the program.
 * We first fetch the site by running the program Ege and Ali written in Python.
 * I use command prompt as a bridge.
 * Then, aiFood.txt will be created.
 * I decode the datas and process them.
 * Then, I provide them to Onur's GUI.
 *
 * Project Members:
 * Denizhan Soydas
 * Onur Kirmizi
 * Sina Sahan
 * Ege Aydin
 * Ali Ozer
 *
 */
public class Launcher {

       public static void main(String[] args) {

              //we are getting data from the file that fetch program created.
              File file = new File("aiFood.txt");
              while(!file.exists()) {
                     System.out.println("Error Loading the site...");

              }




              ArrayList<String> answers = new ArrayList<String>();
              ArrayList<String> oldClues = new ArrayList<String>();
              //ArrayList<String> newClues = new ArrayList<String>();
              ArrayList<Integer> order = new ArrayList<Integer>();
              int downWardStartIndex = -1;
              ArrayList<Coordinate> coordinates = new ArrayList<Coordinate>();
```

```
                Scanner scPuzzle = null;
                try {
                        scPuzzle = new Scanner(file);
                } catch (FileNotFoundException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                        System.out.println("file i/o error on Puzzle");
                }


                if (scPuzzle != null) {
                        String nextWord = " ";

                        // Fetching the order
                        while (nextWord.charAt(nextWord.length() - 1) != ']') {
                                nextWord = scPuzzle.next();
                                String order_not_processed = nextWord;
                                if (order_not_processed.charAt(0) == '[')
                                        order_not_processed = (String)
order_not_processed.subSequence(1, order_not_processed.length());
                                order.add(new Integer((int) (order_not_processed.charAt(0) -
48)));
                        }

                        for (int i = 0; i < order.size(); i++) {
                                System.out.println(order.get(i));
                        }
                        for(int i = 0; i <= order.size() - 2; i++) {
                                if(order.get(i) > order.get(i + 1)) {
                                        downWardStartIndex = i+1;
                                        break;
                                }
                        }
                        System.out.println("DownwardStartIndex is: " + downWardStartIndex);
                        nextWord = " ";
                        // Fetching the answers
                        while (nextWord.charAt(nextWord.length() - 1) != ']') {

                                nextWord = scPuzzle.next();
                                answers.add(nextWord);

                        }

                        // Rendering the answers.
                        answers.set(0, (String) answers.get(0).subSequence(1,
answers.get(1).length() + 1));
                        System.out.println("Size: " + answers.size());
                        answers.set(answers.size() - 1,
                                        (String) answers.get(answers.size() - 1).subSequence(0,
answers.get(1).length()));
                        for (int i = 0; i < answers.size(); i++) {
                                answers.set(i, (String) answers.get(i).subSequence(1,
answers.get(i).length() - 2));
                        }

                        // printing the answers
                        for (int i = 0; i < answers.size(); i++) {
                                System.out.println(answers.get(i));
                        }

                        nextWord = " ";
                        // Fetching the old clues
                        nextWord = scPuzzle.nextLine();
                        nextWord = scPuzzle.nextLine();
                        String[] clues = nextWord.split("',|\",");
                        for (String x : clues) {
                                oldClues.add(x);
                        }
                        for (String x : oldClues) {
                                System.out.println(x);
                        }
                        //now rendering the clues.
                        oldClues.set(0, (String)oldClues.get(0).subSequence(2, 
oldClues.get(0).length()));
                        oldClues.set(oldClues.size()-1, (String)oldClues.get(oldClues.size()-
1).subSequence(0, oldClues.get(oldClues.size()-1).length()-2));
                        for (int i = 1; i<= oldClues.size() -1;i++) {
                                if(oldClues.get(i).charAt(0) == '\"' && 
oldClues.get(i).charAt(oldClues.get(i).length()) == '\"'){
                                        oldClues.set(i,(String)oldClues.get(i).subSequence(1, 
oldClues.get(i).length()-1));
                                }
                                else {
                                        oldClues.set(i, (String)oldClues.get(i).subSequence(2, 
oldClues.get(i).length()));
                                }
                        }
                        for (String x : oldClues) {
                                System.out.println(x);
                        }

                        nextWord = " ";
                        // Fetching the coordinates
                        nextWord = scPuzzle.nextLine();
                        //System.out.println(nextWord);
                        String[] coordinates_not_rendered = nextWord.split(";");
                        for(String x: coordinates_not_rendered) {
                                System.out.println(x);
                        }
                        for(int i = 0; i<= coordinates_not_rendered.length-1;i++) {
                                coordinates_not_rendered[i] = 
(String)coordinates_not_rendered[i].subSequence(1, coordinates_not_rendered[i].length());
                        }
                        coordinates_not_rendered[coordinates_not_rendered.length-1] = 
(String)coordinates_not_rendered[coordinates_not_rendered.length-1].subSequence(0, 
coordinates_not_rendered[coordinates_not_rendered.length-1].length() -1);

                        for(String x: coordinates_not_rendered) {
                                System.out.println(x);
                        }
                        for(String x: coordinates_not_rendered) {
                                Coordinate tempCoordinate = new Coordinate((int)x.charAt(3) - 
48,(int)x.charAt(5) - 48);
```

```
                                coordinates.add(tempCoordinate); // WARNING INDEX STARTS AT 0
BUT, PUZZLE STARTS AT 1st KEY!
                        }
                        for(Coordinate x: coordinates) {
                                System.out.println(x);
                        }

                        Date date;
                        try {
                                date = new SimpleDateFormat("dd/MM/yyyy").parse(args[0]);
                        } catch (ParseException e1) {
                                date = new Date();
                        }
                        String dateTrimmed = ((String)date.toString().subSequence(0, 10) + "," +
(String)date.toString().subSequence(24,date.toString().length()));

                        System.out.println(date);
                        System.out.println(dateTrimmed);
                        System.out.println(order);

                        String preSpace = "       ";
                        String postSpace = "    ";
                        int maxSize = 30;

                        // Creating Area Strings.
                        String acrossAreaOld = "";
                        String downAreaOld = "";
                        for(int i = 0; i < downWardStartIndex; i++) {
                                //acrossAreaOld = acrossAreaOld + "\n" + order.get(i) + "    " +
oldClues.get(i) + "    ";
                                acrossAreaOld = acrossAreaOld + preSpace + order.get(i) +
postSpace;
                                int size = 0;
                                String[] words = oldClues.get(i).split(" ");
                                for(int j = 0; j < words.length; j++) {
                                        size += words[j].length();
                                        acrossAreaOld = acrossAreaOld + words[j] + " ";
                                        if(size > maxSize && words.length > j+1) {
                                                acrossAreaOld = acrossAreaOld + "\n"+ postSpace +
"  " + preSpace;

                                                size = 0;
                                        }
                                }
                                if(acrossAreaOld.charAt(acrossAreaOld.length() - 1) == '\n') {
                                        acrossAreaOld =
acrossAreaOld.substring(0,acrossAreaOld.length() - 1);
                                }

                                acrossAreaOld = acrossAreaOld + "\n";



                        }
                        for(int i = downWardStartIndex; i < order.size(); i++) {
                                //downAreaOld = downAreaOld + "\n" + order.get(i) + "    " +
oldClues.get(i) + "    ";
                                downAreaOld = downAreaOld + preSpace + order.get(i) + postSpace;
                                String[] words = oldClues.get(i).split(" ");
```

```
                int size = 0;
                for(int j = 0; j < words.length; j++) {
                    size += words[j].length();
                    downAreaOld = downAreaOld + words[j] + " ";
                    if(size > maxSize && words.length > j+1) {
                        downAreaOld = downAreaOld + "\n"+ postSpace + "  " 
+ preSpace;

                        size = 0;
                    }
                }
                if(downAreaOld.charAt(downAreaOld.length() - 1) == '\n') {
                    downAreaOld = downAreaOld.substring(0,downAreaOld.length() 
- 1);

                }

                downAreaOld = downAreaOld + "\n";


            }


            System.out.println(acrossAreaOld);
            System.out.println(downAreaOld);

            String[] letters = new String[downWardStartIndex];
            for(int i=0;i<=downWardStartIndex - 1; i++) {
                letters[i] = answers.get(i);
            }
            System.out.println("CHECKPOINT ***");
            for(int i=0; i <= letters.length-1; i++) {
                System.out.println(letters[i]);
            }
            acrossAreaOld = acrossAreaOld.replace("\\\'", "\'");
            downAreaOld = downAreaOld.replace("\\\'", "\'");











            System.out.println("We fetched the whole puzzle");
            System.out.println("Now its time to generate new clues...");





            //we are getting new clues from the file that is generated
            File fileNew = new File("generatedClues.txt");
            while(!fileNew.exists()) {
                System.out.println("Error generating Clues...");
            }
```

```
                     Scanner scClues = null;
                     try {
                            scClues = new Scanner(fileNew);
                     } catch (FileNotFoundException e) {
                            // TODO Auto-generated catch block
                            e.printStackTrace();
                            System.out.println("file i/o error on Clues");
                     }



                     String allNewClues = scClues.nextLine();
                     String[] cluesNew = allNewClues.split("#####"); // REGEX !
                     ArrayList<String> cluesNewAL = new ArrayList<String>();
                     for (String x : cluesNew) {
                            cluesNewAL.add(x.substring(0, 1).toUpperCase() + x.substring(1));
                     }


                     String acrossAreaNew = "";
                     String downAreaNew = "";
                     for(int i = 0; i < downWardStartIndex; i++) {
                            //acrossAreaNew = acrossAreaNew + "\n" + order.get(i) + "    " +
cluesNewAL.get(i) + "    ";
                            acrossAreaNew = acrossAreaNew + preSpace + order.get(i) +
postSpace;
                            String[] words = cluesNewAL.get(i).split(" ");
                            int size = 0;
                            for(int j = 0; j < words.length; j++) {
                                   size += words[j].length();
                                   acrossAreaNew = acrossAreaNew + words[j] + " ";
                                   if(size > maxSize && words.length > j+1) {
                                          acrossAreaNew = acrossAreaNew + "\n"+ postSpace +
"  " + preSpace;

                                          size = 0;
                                   }
                            }
                            if(acrossAreaNew.charAt(acrossAreaNew.length() - 1) == '\n') {
                                   acrossAreaNew =
acrossAreaNew.substring(0,acrossAreaNew.length() - 1);
                            }
                            acrossAreaNew = acrossAreaNew + "\n";
                            acrossAreaNew = acrossAreaNew.substring(0, 1).toUpperCase() +
acrossAreaNew.substring(1);
                     }
                     for(int i = downWardStartIndex; i < order.size(); i++) {
                            //downAreaNew = downAreaNew + "\n" + order.get(i) + "    " +
cluesNewAL.get(i) + "    ";
                            downAreaNew = downAreaNew + preSpace + order.get(i) + postSpace;
                            String[] words = cluesNewAL.get(i).split(" ");
                            int size = 0;
                            for(int j = 0; j < words.length; j++) {
                                   size += words[j].length();
                                   downAreaNew = downAreaNew + words[j] + " ";
                                   if(size > maxSize && words.length > j+1) {
                                          downAreaNew = downAreaNew + "\n"+ postSpace + "  "
+ preSpace;
```

```
                                                size = 0;
                                        }
                                }
                                if(downAreaNew.charAt(downAreaNew.length() - 1) == '\n') {
                                        downAreaNew = downAreaNew.substring(0,downAreaNew.length()
- 1);
                                }

                                downAreaNew = downAreaNew + "\n";
                                downAreaNew = downAreaNew.substring(0, 1).toUpperCase() +
downAreaNew.substring(1);


                        }
                        acrossAreaOld = acrossAreaOld.replace("&#128514;", ":)");// WARNING
                        downAreaOld = downAreaOld.replace("&#128514;", ":)");// WARNING
                        GUICreator gc = new GUICreator(acrossAreaOld, downAreaOld,
acrossAreaNew, downAreaNew, dateTrimmed, letters, coordinates);


                        //Saying Goodbye.
                        scPuzzle.close();
                        scClues.close();
                }


        }



}
```

GUICreator.java

```
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.GridLayout;
import java.util.ArrayList;

import javax.swing.BorderFactory;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.border.Border;

public class GUICreator {
        JFrame frame;
        GamePanel myGamePanel;
        JTextArea oldCluesAcross;
        JTextArea oldCluesDowns;
        JTextArea newCluesAcross;
        JTextArea newCluesDowns;
        Border border2;
        JPanel upperPanel;
        JPanel rightPanel;
        JPanel lowerPanel;
        JPanel rightGrid;
        JTextField date;
        JTextField infos;
        String[] lettersAdjusted;

        // added in v1.3
        JPanel AcUppPanel;
        JPanel DoUppPanel;
        JPanel AcLowPanel;
        JPanel DoLowPanel;

        JPanel AcUppPanelH;
        JPanel DoUppPanelH;
        JPanel AcLowPanelH;
        JPanel DoLowPanelH;

        public GUICreator(
                        String cluesAcrossOld,
                        String cluesDownOld,
                        String cluesAcrossNew,
                        String cluesDownNew,
                        String date1,
                        String[] letters,
                        ArrayList<Coordinate> coordinatesOfNumbers
                        ) {
                System.out.println(letters.length);
                lettersAdjusted = new String[letters.length*letters.length];
                for(int i = 0 ; i<=letters.length-1; i++ ) {
                        for(int j = 0 ; j <= letters.length-1 ; j++ ) {

                                lettersAdjusted[convertCoordinates(i,j)] =
String.valueOf(letters[i].charAt(j));

                        }
                }
                AcUppPanel = new JPanel(new BorderLayout());
                DoUppPanel = new JPanel(new BorderLayout());
                AcLowPanel = new JPanel(new BorderLayout());
                DoLowPanel = new JPanel(new BorderLayout());

                AcUppPanelH = new JPanel(new BorderLayout());
                DoUppPanelH = new JPanel(new BorderLayout());
                AcLowPanelH = new JPanel(new BorderLayout());
                DoLowPanelH = new JPanel(new BorderLayout());


                frame = new JFrame();
                infos = new JTextField(date1 +"    " + "Zero IQ AI");
                JPanel bottomP = new JPanel(new BorderLayout());
                JPanel bottomP2 = new JPanel(new BorderLayout());
                //bottomP2.setSize(700, 700);
                infos.setHorizontalAlignment(JTextField.RIGHT);
                infos.setFont(new Font("Arial", Font.PLAIN, 15));
                infos.setEditable(false);
                infos.setBorder(null);
```

```
bottomP2.add(infos, BorderLayout.EAST);
bottomP2.setPreferredSize(new Dimension(680,20));
bottomP.add(bottomP2, BorderLayout.WEST);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
myGamePanel = new GamePanel();

oldCluesAcross = new JTextArea(15, 15); //!setSize WARNING
oldCluesDowns = new JTextArea(15, 15); //!setSize WARNING

newCluesAcross = new JTextArea(15, 15); //!setSize WARNING
newCluesDowns = new JTextArea(15, 15); //!setSize WARNING


oldCluesAcross.setText("\n"+cluesAcrossOld);
oldCluesDowns.setText("\n"+cluesDownOld);

newCluesAcross.setText("\n"+cluesAcrossNew);
newCluesDowns.setText("\n"+cluesDownNew);

oldCluesAcross.setEditable(false);
oldCluesDowns.setEditable(false);

newCluesAcross.setEditable(false);
newCluesDowns.setEditable(false);
JTextField AcUppLabel = new JTextField("       Across");
JTextField DoUppLabel = new JTextField("       Down");
JTextField AcLowLabel = new JTextField("       New Across");
JTextField DoLowLabel = new JTextField("       New Down");

AcUppLabel.setBorder(null);
DoUppLabel.setBorder(null);
AcLowLabel.setBorder(null);
DoLowLabel.setBorder(null);


Font f = new Font (Font.DIALOG, Font.BOLD, 14);
Font f2 = new Font (Font.DIALOG, Font.PLAIN, 13);
AcUppLabel.setFont(f);
DoUppLabel.setFont(f);
AcLowLabel.setFont(f);
DoLowLabel.setFont(f);
oldCluesAcross.setFont(f2);
oldCluesDowns.setFont(f2);
newCluesAcross.setFont(f2);
newCluesDowns.setFont(f2);

AcUppPanelH.setBackground(Color.white);
DoUppPanelH.setBackground(Color.white);
AcLowPanelH.setBackground(Color.white);
DoLowPanelH.setBackground(Color.white);

AcUppPanelH.add(AcUppLabel,BorderLayout.SOUTH);
DoUppPanelH.add(DoUppLabel,BorderLayout.SOUTH);
AcLowPanelH.add(AcLowLabel,BorderLayout.SOUTH);
DoLowPanelH.add(DoLowLabel,BorderLayout.SOUTH);

AcUppPanel.add(AcUppPanelH,BorderLayout.CENTER);
```

```
DoUppPanel.add(DoUppPanelH,BorderLayout.CENTER);
AcLowPanel.add(AcLowPanelH,BorderLayout.CENTER);
DoLowPanel.add(DoLowPanelH,BorderLayout.CENTER);


AcUppPanel.add(oldCluesAcross,BorderLayout.SOUTH);
DoUppPanel.add(oldCluesDowns,BorderLayout.SOUTH);
AcLowPanel.add(newCluesAcross,BorderLayout.SOUTH);
DoLowPanel.add(newCluesDowns,BorderLayout.SOUTH);




AcUppPanel.setVisible(true);
DoUppPanel.setVisible(true);
AcLowPanel.setVisible(true);
DoLowPanel.setVisible(true);



rightPanel = new JPanel(new BorderLayout());
rightGrid = new JPanel(new GridLayout(2,2));
upperPanel = new JPanel(new BorderLayout());
lowerPanel = new JPanel(new BorderLayout());


rightPanel.setBackground(Color.blue);
upperPanel.setOpaque(false); // WARNING ********
lowerPanel.setOpaque(true); // WARNING ********

rightGrid.add(AcUppPanel, BorderLayout.EAST);
rightGrid.add(DoUppPanel, BorderLayout.CENTER);
rightGrid.add(AcLowPanel, BorderLayout.CENTER);
rightGrid.add(DoLowPanel, BorderLayout.CENTER);

date = new JTextField();

date.setText(date1);
date.setHorizontalAlignment(JTextField.LEFT);
date.setEditable(false);
date.setBackground(Color.WHITE);

rightPanel.add(rightGrid);

upperPanel.setVisible(true);
lowerPanel.setVisible(true);
rightGrid.setVisible(true);
rightPanel.setVisible(true);

frame.add(myGamePanel.main_panel, BorderLayout.CENTER);
frame.add(rightPanel, BorderLayout.EAST);
frame.add(bottomP, BorderLayout.SOUTH);

frame.setVisible(true);


for(int i = 0 ; i<coordinatesOfNumbers.size(); i++) {
```

```
            int x = coordinatesOfNumbers.get(i).getX();
            int y = coordinatesOfNumbers.get(i).getY();

            myGamePanel.adjustUpperNumber(i+1,convertCoordinates(x,y));
        }


    for(int i = 0 ; i<25;i++) {
            for(int j = 0 ;j < coordinatesOfNumbers.size(); j++) {

                int convertedCoordinate = 
convertCoordinates(coordinatesOfNumbers.get(j).getX(), coordinatesOfNumbers.get(j).getY());

                if( i == convertedCoordinate) {
                    break;
                }
                if(j == coordinatesOfNumbers.size() - 1) {
                    myGamePanel.adjustUpperNumber(0,i);
                }
            }
        }

    for(int i = 0 ; i<25 ; i++) {
        myGamePanel.enterLetter(lettersAdjusted[i], i );


    frame.setSize(700 + rightPanel.getWidth(), 700);


    frame.setSize(frame.getWidth(), frame.getWidth() - rightPanel.getWidth()  + 
infos.getHeight());


    frame.setResizable(true);
    frame.getContentPane().revalidate();

}

    public int convertCoordinates(int x, int y) { // converts 2D coordinates to 1D 
suitable for panels array
        int result = x * 5 + y;
        return result;
    }

    public int getMaxWidthOfRightPanel() {
        return Math.max(upperPanel.getWidth(), lowerPanel.getWidth());
    }

    public int getMaxHeightOfRightPanel() {
        return Math.max(upperPanel.getHeight(), lowerPanel.getHeight());
    }

    public int getTotalHeight() {
        return upperPanel.getHeight() + lowerPanel.getHeight();
    }

    public int getAcrossWidthMax() {
```

```
		return Math.max(oldCluesAcross.getWidth(), newCluesAcross.getWidth());
	}

	public int getDownWidthMax() {
		return Math.max(oldCluesDowns.getWidth(), newCluesDowns.getWidth());
	}

}
```

GamePanel.java

```
import java.awt.*;
import javax.swing.*;
import javax.swing.border.Border;

public class GamePanel {
	JPanel main_panel;
	JPanel[] panels;
	Border border;
	// JTextField groupName;

	GamePanel() {
		main_panel = new JPanel();
		main_panel.setBackground(Color.white);
		panels = new JPanel[25];
		border = BorderFactory.createLineBorder(Color.BLACK, 1);

		for (int i = 0; i < 25; i++) {
			GridLayout gridLayout1 = new GridLayout(3, 3);
			panels[i] = new JPanel(gridLayout1);
			panels[i].setBackground(Color.WHITE);
			panels[i].setBorder(border);
			panels[i].setVisible(true);
			main_panel.add(panels[i]);
		}

		GridLayout gridLayout = new GridLayout(5, 5);
		main_panel.setLayout(gridLayout);
		main_panel.setVisible(true);
	}

	int labelSize = 20;

	public void adjustUpperNumber(int num, int index) {

		if (num == 0) {

			JLabel label1 = new JLabel("");
			label1.setFont(new Font("Arial", Font.PLAIN, labelSize));
			label1.setHorizontalAlignment(SwingConstants.CENTER);
			label1.setVerticalAlignment(SwingConstants.CENTER);

			panels[index].add(label1);

			JLabel label2 = new JLabel("");
			label2.setFont(new Font("Arial", Font.PLAIN, labelSize));
			label2.setHorizontalAlignment(SwingConstants.CENTER);
```

```
        label2.setVerticalAlignment(SwingConstants.CENTER);

        panels[index].add(label2);

        JLabel label3 = new JLabel("");
        label3.setFont(new Font("Arial", Font.PLAIN, labelSize));
        label3.setHorizontalAlignment(SwingConstants.CENTER);
        label3.setVerticalAlignment(SwingConstants.CENTER);

        panels[index].add(label3);

        JLabel label4 = new JLabel("");
        label4.setFont(new Font("Arial", Font.PLAIN, labelSize));
        label4.setHorizontalAlignment(SwingConstants.CENTER);
        label4.setVerticalAlignment(SwingConstants.CENTER);

        panels[index].add(label4);
    } else {

        JLabel label5 = new JLabel(Integer.toString(num));
        label5.setFont(new Font("Arial", Font.PLAIN, labelSize));
        label5.setForeground(Color.BLACK);

        label5.setHorizontalAlignment(SwingConstants.CENTER);
        label5.setVerticalAlignment(SwingConstants.CENTER);

        panels[index].add(label5);

        JLabel label6 = new JLabel("");
        label6.setHorizontalAlignment(SwingConstants.CENTER);
        label6.setVerticalAlignment(SwingConstants.CENTER);

        panels[index].add(label6);

        JLabel label7 = new JLabel("");
        label7.setHorizontalAlignment(SwingConstants.CENTER);
        label7.setVerticalAlignment(SwingConstants.CENTER);

        panels[index].add(label7);

        JLabel label8 = new JLabel("");
        label8.setHorizontalAlignment(SwingConstants.CENTER);
        label8.setVerticalAlignment(SwingConstants.CENTER);

        panels[index].add(label8);

    }
}

public void enterLetter(String letter, int index) {

    if (letter.charAt(0) == '#') {
        blackCell(index);
    } else {

        JLabel label = new JLabel(letter);
        label.setFont(new Font("Arial", Font.PLAIN, 40));
				label.setHorizontalAlignment(SwingConstants.CENTER);
				label.setVerticalAlignment(SwingConstants.CENTER);

				panels[index].add(label);

				JLabel label2 = new JLabel("");
				label2.setHorizontalAlignment(SwingConstants.CENTER);
				label2.setVerticalAlignment(SwingConstants.CENTER);

				panels[index].add(label2);

				JLabel label3 = new JLabel("");
				label3.setHorizontalAlignment(SwingConstants.CENTER);
				label3.setVerticalAlignment(SwingConstants.CENTER);

				panels[index].add(label3);
			}
		}

		public void blackCell(int index) {
			panels[index].setBackground(Color.BLACK);
		}

	}
```

Coordinate.java

```
/**
 *
 * @author Denizhan
 * This Class is an object form simulator for Coordinates.
 *
 */
public class Coordinate {
	int x;
	int y;

	Coordinate(int x, int y) {
		this.x = x;
		this.y = y;

	}

	int getX() {
		return x;

	}

	int getY() {
		return y;
	}

	void setX(int x) {
		this.x = x;
	}

	void setY(int y) {
		this.y = y;
```

```
        }

        public String toString() {
                return "" + x + "," + y;
        }

}
```

Appendix C: Clue generator written in Python

```
#!/usr/bin/env python
# coding: utf-8
"""
Created on Sun Dec 8 11:56:54 2019

@author: Ege Aydin
"""

# # CS461 Project

# ## Imports


print("ZERO IQ AI")
print("Importing necessary libraries.")
import json
import argparse
import urllib.request
import re
from urllib.parse import quote
import bs4 as bs
import nltk
from gensim.models import Word2Vec
import numpy as np
from nltk.stem import PorterStemmer
from nltk.corpus import stopwords
from gensim.models import KeyedVectors
from html import unescape
import torch
from pytorch_pretrained_bert import BertTokenizer, BertModel, BertForMaskedLM

def cleanString(str):
    ans = str.lower()
    ans = re.sub('[^a-zA-Z]', ' ', ans )
    ans = re.sub(r'\s+', ' ', ans)
    return ans
#cleanString("sda adsdsa")


# ## Merriam-Webster Scraper
# Merriam Webster API code modified from the code that is retrieved from
# https://github.com/fluencer/dictionary-cli-app/blob/master/merriam.py

keyfile1 = open("collegiate.txt", 'r')
keyfile2 = open("learner.txt", 'r')
keys1 = keyfile1.readline()
keys2 = keyfile2.readline()
keyfile1.close()
```

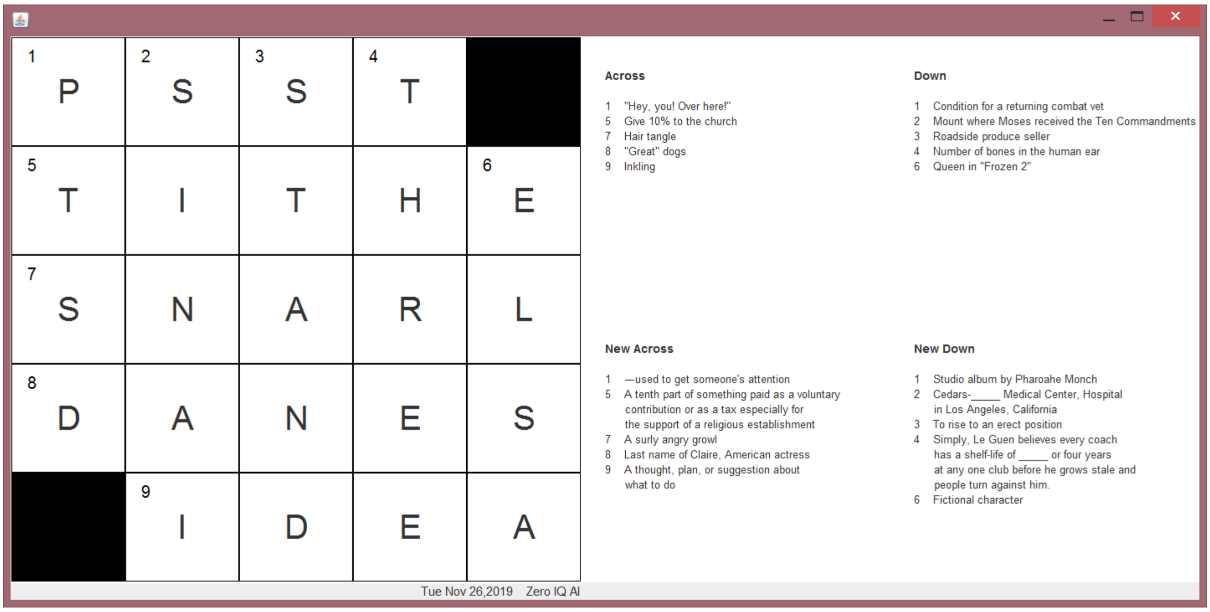
```
keyfile2.close()

def dictionarySearch(query):
    query = query.lower()
    urlfrmt1 = "https://dictionaryapi.com/api/v3/references/collegiate/json/"+\
 quote(query)+"?key=" + keys1
    urlfrmt2 = "https://dictionaryapi.com/api/v3/references/learners/json/"+\
 quote(query)+"?key=" + keys2
    response1 = urllib.request.urlopen(urlfrmt1)
    jsStruct1 = json.load(response1)
    response2 = urllib.request.urlopen(urlfrmt2)
    jsStruct2 = json.load(response2)

    all_definitions = []
    for meaning in jsStruct1+jsStruct2:
        try:
            word = meaning['meta']['id']
            if (word.count(':') != 0 and word[:word.find(':')].lower() == query) or \
 word.lower() == query:
                #print("I am here")
                definitions = meaning['shortdef']
                all_definitions += definitions
        except TypeError:
            break
        except KeyError:
            pass

    ps = PorterStemmer()
    all_definitions = [e for e in all_definitions if not query in e]
    all_definitions = [e for e in all_definitions if not ps.stem(query) in e]

    examples = []

    for usage in jsStruct1+jsStruct2:
        try:
            longdefs = usage['def']
            for i, eachDef in enumerate(longdefs, 1):
                if 'sseq' in eachDef:
                    # test[*][0][1]['dt'] - gives list of illustrations
                    for ill in eachDef['sseq']:
                        if 'dt' in ill[0][1]:
                            for illus in ill[0][1]['dt']:
                                if illus[0] == 'vis':
                                    # enumerate
                                    for ii in range(1, len(illus)):
                                        L = len(illus[ii])
                                        for jj in range(L):
                                            if 't' in illus[ii][jj]:
                                                text = re.sub(r"{.*?}", "",\
     illus[ii][jj]['t'])
                                                if(text.count(query) == 1):
                                                    examples.append(text)
        except KeyError:
            pass
        except TypeError:
            break
    included_words = []
    for sentence in examples:
        if len(sentence.split()) == 1: # if it's a word
            included_words.append(sentence)

    for sentence in included_words:
        examples.remove(sentence)

    print(query.upper(), " : Definitions = ", all_definitions, "\n\n" + \
query.upper(), " : Example Sentences = ", examples, "\n\n" +     \
query.upper(), " : Words that include the answer = ", included_words, "\n\n")
    return (all_definitions, examples, included_words)

#x = dictionarySearch("blue") #(definitions, examples)


# ## Clue Comparator

# To make sure the clues are changed

def compareSentences(query, clue, sentence):
    ps = PorterStemmer()
    keywords = [cleanString(sentence).split(), cleanString(clue).split()]
    keywords[0] = [ps.stem(word) for word in keywords[0] if not word in \
    stopwords.words('english')]
    keywords[0] = np.unique(keywords[0])
    keywords[1] = [ps.stem(word) for word in keywords[1] if not word in \
    stopwords.words('english')]
    keywords[1] = np.unique(keywords[1])
    intersection = len([word for word in keywords[1] if word in keywords[0]])
    if(min(len(keywords[0]),len(keywords[1])) == 0):
        return 1
    ans = intersection / min(len(keywords[0]),len(keywords[1])) < 0.5
    if not ans:
        print(query.upper(), " : Too similar to the old clue. Passing ("+sentence+")")
    return ans

#compareSentences("Ring-shaped reef", "an island that is made of coral and shaped like
# a ring")


# ## Example Sentence Scraper

def scrapeWordHippo(query):
    scrapped_data = urllib.request.urlopen(\
'https://www.wordhippo.com/what-is/sentences-with-the-word/' + \
quote(query.lower())+'.html')
    article = scrapped_data.read()

    parsed_article = bs.BeautifulSoup(article,'lxml')

    ans = []
    for e in parsed_article.find('table', id='mainsentencestable').findAll(\
'td', id=re.compile(r'exv2st.*')):
        ans.append(e.get_text())
    ps = PorterStemmer()
    ans = [sentence for sentence in ans if sentence.count(ps.stem(query)) == 1]
    # don't take if there are more than 1 query, it'll confuse BERT
    if len(ans) > 1:
        print(query.upper(), " : More Examples = [", ans[0], ",", ans[1], ", ...]\n\n")
    else:
        print(query.upper(), " : More Examples = ", ans, "\n\n")
    return ans

#examples = scrapeWordHippo("blue")


# ## Example Sentence Picker Using BERT

# load pretrained bert model
print("Loading pretrained neural network BERT.")
bert_model = BertForMaskedLM.from_pretrained('bert-base-uncased')
bert_model.eval()
print("Model loaded")


def getProbability(text, word):
    text = text.replace("[=", ". ").replace("]", "")

    text = "[CLS] " + re.sub(r"[^ ]*"+word+r"[^ ]*", "[MASK]", text) + " [SEP]"
    if(text == "[CLS] [MASK] [SEP]"):
        return 0
    # Load pre-trained model tokenizer (vocabulary)
    tokenizer = BertTokenizer.from_pretrained('bert-base-uncased', do_lower_case=True,\
  do_basic_tokenize=True)
    tokenized_text = tokenizer.tokenize(text)
    indexed_tokens = tokenizer.convert_tokens_to_ids(tokenized_text)

    # Create the segments tensors.
    segments_ids = [0] * len(tokenized_text)

    # Convert inputs to PyTorch tensors
    tokens_tensor = torch.tensor([indexed_tokens])
    segments_tensors = torch.tensor([segments_ids])

    # Predict all tokens
    with torch.no_grad():
        predictions = bert_model(tokens_tensor, segments_tensors)

     masked_index = 0
     try:
         masked_index = tokenized_text.index('[MASK]')
     except:
         return 0
     pp = torch.nn.functional.softmax(predictions, -1)
#    (prob, item) = torch.topk(pp[0, masked_index],k=1000)
#    top_ten = []
#    for i in range(len(item)):
#        top_ten.append((tokenizer.convert_ids_to_tokens([item[i].item()])[0],
float(prob[i])))
#    print(top_ten)
    prob = 0.0
    for e in tokenizer.convert_tokens_to_ids(tokenizer.tokenize(word)):
        prob = max(pp[0,masked_index][e], prob)
    return float(prob)
```

```
#print(getProbability("Beetroot is also great for making soup and can be sliced, diced or
#grated to add vivid
#colour to salads.", "diced"))


# His reward was the monastery's sanctus bell which had a blue clapper.
# His reward was the monastery's sanctus bell which had a [MASK] clapper.
# red %7
# blue %6


def getBestSentence(examples, word, clue):
    best_sentence = ""
    best_probability = 0
    if(" " in word):
        return("", 0)
    #print(examples)
    for e in examples:
        if len(e.split())==1:
            continue
        if not compareSentences(word, clue, e):
            continue
        ps = PorterStemmer()
        if e.lower().count(ps.stem(word)) != 1:
            continue
        print(word.upper(), " : Evaluating example sentence ("+e+")")
        prob = getProbability(e, word)
        print(word.upper(), " : Probability of guessing the word: " + str(int(prob*100))\
+ "%\n\n")


        if prob > best_probability:
            best_sentence = e
            best_probability = prob

        if best_probability > 0.99:
            break
    return (best_sentence.replace(word, "_____"), best_probability)


#getBestSentence(x[1]+examples, "blue", "")


# ## Google Knowledge Graph API
# A modified version of python wrapper for Google Knowledge Graph API
# that is retrieved from https://developers.google.com/knowledge-graph
# API key is removed from the code.

def googleKnowledgeGraph(query):
    api_key = "ENTER KEY HERE"
    service_url = 'https://kgsearch.googleapis.com/v1/entities:search'
    params = {
        'query': query,
        'limit': 10,
        'indent': True,
        'key': api_key,
    }
    url = service_url + '?' + urllib.parse.urlencode(params)
    response = json.loads(urllib.request.urlopen(url).read())
    #print(response['itemListElement'])
    meanings = []
    ps = PorterStemmer()
    for element in response['itemListElement']:
        name = unescape(element['result']['name'])
        try:
            details = unescape(\
     element['result']['detailedDescription']['articleBody']).split(".")[0]
            URL = unescape(element['result']['detailedDescription']['url'])
        except KeyError:
            details = ''
            URL = ''
            pass
        resultType = unescape(element['result']['@type'])
        if name.lower().count(query) == 1:
            s_name = ''
            if name.lower() != query.lower():
                if "Person" in resultType and name.split()[-1].lower() == query:
                    s_name = "Last name of " + re.sub(r"(?i) " + query, "", name)+", "
                elif "Person" in resultType and name.split()[0].lower() == query:
                    s_name = "First name of "+re.sub(r"(?i)" + query + " ", "", name)+", "
                else:
                    s_name = re.sub(r"(?i)" + query, "_____", name) + ", "
            try:
                meaning = s_name + unescape(element['result']['description'])
                if not query in meaning.lower() and not ps.stem(query) in meaning.lower():
                    meanings.append(meaning)
            except KeyError:
                pass
            #print(details.lower()[:details.find("is")])
            s_details = ''
            if details.lower()[:details.find("is")-1] == name.lower():
                s_details = (details[details.index("is")+3:])
            elif details.lower().count(query) == 1:
                s_details = (re.sub(r"(?i)" + query, "_____", details))
            elif query not in details.lower():
                s_details = (details)
            if("_____" in s_name and "_____" in s_details):
                if not query in s_details.lower() and not ps.stem(query) in\
  s_details.lower():
                    meanings.append(s_details)
            elif details != '':
                if not query in (s_name + s_details).lower() and not ps.stem(query) in\
 (s_name + s_details).lower():
                    meanings.append(s_name + s_details)
            #print(URL, "\n")
    print(query.upper(), " : Data from Google Knowledge Graph = ", meanings,"\n\n")
    return meanings

#googleKnowledgeGraph("dys")


# ## Definition Sentence Picker
# Compares the sum of sentence vector with the original word using cosine similarity


# load pretrained word vectors
print("Loading pretrained word vector dataset.")
word_vector_model = KeyedVectors.load_word2vec_format(\
     'GoogleNews-vectors-negative300.bin', binary=True, limit=10 ** 6)
print("Dataset loaded.")
# 8GB ram wasn't enough to load all. Delete limit if ram>8GB



def findBestDefinition(query, definition_list, clue):
    best_definition = ""
    best_similarity = 0

    query_words = query.split()
    if not query in stopwords.words('english'):
        query_words = [word for word in query_words if not word in\
stopwords.words('english')]

    if definition_list == []:
        print(query.upper(), " : No definition available to evaluate.")
        return (best_definition, best_similarity)

    for word in query_words:
        if not word in word_vector_model:
            print(query.upper(), " : Word isn't in the dataset. Cannot evaluate definitions.")
            best_definition = max(definition_list, key=len)
            best_similarity = -1
            return (best_definition, best_similarity)

    for sentence in definition_list:
        if not compareSentences(query, clue, sentence):
            continue
        ps = PorterStemmer()
        if ps.stem(query) in sentence.lower():
            continue

        tabooF = open("taboo.txt")
        taboo = tabooF.readlines()
        tabooF.close()
        if any(word in sentence.lower() for word in taboo):
            continue
        print(query.upper(), " : Evaluating definition ("+sentence+")")

        search_words = cleanString(sentence).split()

        sentence_clear = [word for word in search_words if not word in
stopwords.words('english')]
        sentence_clear = np.unique(sentence_clear)
        total_words = len(sentence_clear)
        sentence_clear = [word for word in sentence_clear if word in
word_vector_model.vocab.keys()]
        if(total_words == 0):
            return ('', 0)
        percentage = len(sentence_clear) / total_words #if all the words aren't present in
vocab, penalize
        u = np.zeros(300)

        for word in sentence_clear:
            u += word_vector_model[word]
        v = np.zeros(300)
        for word in query_words:
            v += word_vector_model[word]
        if(np.linalg.norm(u) == 0.0):
            print(query.upper(), " : Failed to evaluate definition\n\n")
            continue
        similarity = percentage*np.dot(u, v)/np.linalg.norm(u)/np.linalg.norm(v)
        if best_similarity < similarity:
            best_definition = sentence
            best_similarity = similarity
        print(query.upper(), " : Similarity is: "+str(int(similarity*100))+"\n\n")
    best_definition = re.sub(r" \(.*\)", "", best_definition)
    return (best_definition, best_similarity)


#findBestDefinition("kind of", x[0])
#s = ['Song by Cool Hand Luke', "First name of Fishman, Hal Fishman's wife", "Noel '_____'
#Minor, Fictional character"]
#findBestDefinition("wang", s)


# ## Prefix Suffix Finder

def prefixSuffixFinder(example_words, query, clue):
    total_results = []
    for word in example_words:
        if(word.index(query) == 0):
            if not word[len(query):] in clue:
                total_results.append("Prefix with -" + word[len(query):])
        elif(word.index(query) == len(word) - len(query)):
            if not word[:word.index(query)] in clue:
                total_results.append("Suffix with " + word[:word.index(query)] + "-")
    return total_results

#prefixSuffixFinder(['dyslexia', 'dysplasia', 'dysphagia', 'dysfunction', 'dyslogistic'],
#"dys", "Prefix with -lexia")


# # Main Wrapper

def getBestChoice(query, clue):
    meaning_threshold = 0
    example_threshold = 0.5
    GKG_threshold = 0.1

    ans = []
    print(query.upper(), " : Getting dictionary data")
    dictionary_data = dictionarySearch(query)
    print(query.upper(), " : Evaluating definitions")
    best_def = findBestDefinition(query, dictionary_data[0], clue)
    if best_def[0] != '' and best_def[1] > meaning_threshold:
        print(query.upper(), " : Best definition is selected as clue (" + best_def[0] + \
 ")\n\n\n---------------------------------\n\n\n")
        return best_def[0]

    print(query.upper(), " : Scraping example sentences")
    examples_word_hippo = []
    try:
        examples_word_hippo = scrapeWordHippo(query)
    except AttributeError:
        pass

    examples = examples_word_hippo + dictionary_data[1]
    print(query.upper(), " : Evaluating example sentences")
    best_example = getBestSentence(examples, query, clue)
    #print(best_example)
    #print("best example selected")

    if best_example[0] != '' and best_example[1] > example_threshold:
        print(query.upper(), " : Most guessable example sentence is selected as clue (" +\
best_example[0] + ")\n\n\n---------------------------------\n\n\n")
        return best_example[0]
    #print("google knowledge graph API")
    print(query.upper(), " : Scraping Google Knowledge Graph")
    google_knowledge = googleKnowledgeGraph(query)
    print(query.upper(), " : Evaluating data")
    best_knowledge = findBestDefinition(query, google_knowledge, clue)

    if best_knowledge[0] != '' and best_knowledge[1] > GKG_threshold:
        print(query.upper()," : Best Google Knowledge Graph output is selected as clue ("\
              +google_knowledge[0] + ")\n\n\n---------------------------------\n\n\n")
        return(google_knowledge[0])

    prefix_suffix = prefixSuffixFinder(dictionary_data[2], query, clue)
    if prefix_suffix != [] and compareSentences(query, clue, prefix_suffix[0]):
        print(query.upper(), " : Word will be described as a part of a bigger word (" +\
prefix_suffix[0] +")\n\n\n---------------------------------\n\n\n")
        return prefix_suffix[0]

    if best_knowledge[0] != '' and best_knowledge[1] == -1 and compareSentences(query,\
clue,google_knowledge[0]):
        print(query.upper(), " : A random Google Knowledge Graph output is selected\
 as clue (" +  google_knowledge[0] + ")\n\n\n---------------------------------\n\n\n")
        return(google_knowledge[0])
    return ''

#getBestChoice("notre", "") # meaning similarity 2 word


def getBestChoice2(word, clue):
    ans = getBestChoice(word,clue)
    if ans != '':
        return ans
    print(word.upper(), " : Clue couldn't be generated. Trying to evaluate the\
 stem of the word.")
    ps = PorterStemmer()
    ans = getBestChoice(ps.stem(word),clue)
    if ans != '':
        print(word.upper(), " : Using the stem of the word to decribe it.")
        return "Derived from the word that is decribed by the following: " + ans

    print(word.upper(), " : Clue couldn't be generated from the stem of the word\
 either. Checking whether the word is actually multiple words combined.")

    url = "https://api.datamuse.com/sug?s=" + quote(word)
    response = json.loads(urllib.request.urlopen(url).read())
    for e in response:
        try:
            if ''.join(e["word"].split()) == word and e["word"] != word:
                print(word.upper(), " : Word is changed into:", e["word"])
                return getBestChoice(e["word"],clue)
        except KeyError:
            pass

    print(word.upper(), " : Word couldn't be divided or no definition found.\
 Trying to use a rhyming word for the clue")


    url = "https://api.datamuse.com/words?rel_rhy=" + quote(word)
    response = json.loads(urllib.request.urlopen(url).read())
    try:
        print(word.upper(), " : Found a rhyming word (" +\
response[0]["word"]+")\n\n\n--------------------------------\n\n\n")
        return '"It rhymes with '+ response[0]["word"] +'," for this answer'
    except KeyError:
        pass

    print(word.upper(), " : Clue generation failed.")

    return 'Failed to generate clue'

#getBestChoice2("notre", "University of ___ Dame")

puzzle = open("aiFood.txt")
text = puzzle.readlines()
puzzle.close()
words = re.sub(r"[^A-Za-z,]", '', text[1]).split(",")
#print(words)
clues = text[2][2:-3].replace("', '", '", "').replace("', \"", '", "').replace("\", '",\
 '", "').split('", "')
for i in range(10):
    clues[i] = clues[i]
#print(clues)
#words = ['DIP', 'VIRAL', 'ECOLI', 'TENET', 'DYS', 'DICED', 'IRONY', 'PALES', 'VET',\
#         'LIT']
new_clues = []
print("Starting clue generation.")
for i in range(10):
    word = words[i].lower()
    clue = clues[i]
    new_clues.append(getBestChoice2(word,clue))

for i in range(10):
    print(words[i].upper() + " :")
    print("\tOld clue: " + clues[i])
    print("\tNew clue: " + new_clues[i])
    print("\n")
file1 = open("generatedClues.txt","w+")
file1.write("#####".join(new_clues))
file1.close()
#np.savetxt("generatedClues.txt", new_clues, delimiter=",", fmt="%s")
print("Clue generation completed.")
input()
```

Appendix D: Sample Output of the Clue Generator

Go to https://drive.google.com/drive/folders/11ExdShgDvhxhlB4glhrfJn7uB9qeg7en?usp=sharing for the output of all 12 sample runs.

NY Times Mini Puzzle (26.11.2019)



1 P	2 S	3 S	4 T	
5 T	I	T	H	6 E
7 S	N	A	R	L
8 D	A	N	E	S
	9 I	D	E	A

Across

1 "Hey, you! Over here!"
5 Give 10% to the church
7 Hair tangle
8 "Great" dogs
9 Inkling

Down

1 Condition for a returning combat vet
2 Mount where Moses received the Ten Commandments
3 Roadside produce seller
4 Number of bones in the human ear
6 Queen in "Frozen 2"

New Across

1 —used to get someone's attention
5 A tenth part of something paid as a voluntary contribution or as a tax especially for the support of a religious establishment
7 A surly angry growl
8 Last name of Claire, American actress
9 A thought, plan, or suggestion about what to do

New Down

1 Studio album by Pharoahe Monch
2 Cedars-_____ Medical Center, Hospital in Los Angeles, California
3 To rise to an erect position
4 Simply, Le Guen believes every coach has a shelf-life of _____ or four years at any one club before he grows stale and people turn against him.
6 Fictional character

Tue Nov 26,2019 Zero IQ AI

```
ZERO IQ AI
Importing necessary libraries.
Loading pretrained neural network BERT.
Model loaded
Loading pretrained word vector dataset.
Dataset loaded.
Starting clue generation.
PSST  : Getting dictionary data
PSST  : Definitions =  ["—used to get someone's attention", "—used to get someone's
attention"]

PSST  : Example Sentences =  []

PSST  : Words that include the answer =  []


PSST  : Evaluating definitions
PSST  : Evaluating definition (—used to get someone's attention)
PSST  : Similarity is: 19
PSST  : Evaluating definition (—used to get someone's attention)
PSST  : Similarity is: 19


PSST  : Best definition is selected as clue (—used to get someone's attention)


---------------------------------




TITHE  : Getting dictionary data
TITHE  : Definitions =  ['a tenth part of something paid as a voluntary contribution or as a
tax especially for the support of a religious establishment', 'tenth; broadly : a small part',
'to pay or give a tenth part of especially for the support of a religious establishment or
organization', "an amount of money that a person gives to a church which is usually equal to
1/10 of that person's income"]

TITHE  : Example Sentences =  []

TITHE  : Words that include the answer =  []


TITHE  : Evaluating definitions
TITHE  : Evaluating definition (a tenth part of something paid as a voluntary contribution or
as a tax especially for the support of a religious establishment)
TITHE  : Similarity is: 41


TITHE  : Evaluating definition (tenth; broadly : a small part)
TITHE  : Similarity is: 14


TITHE  : Too similar to the old clue. Passing (to pay or give a tenth part of especially for
the support of a religious establishment or organization)
TITHE  : Too similar to the old clue. Passing (an amount of money that a person gives to a
church which is usually equal to 1/10 of that person's income)
TITHE  : Best definition is selected as clue (a tenth part of something paid as a voluntary
contribution or as a tax especially for the support of a religious establishment)


---------------------------------




SNARL  : Getting dictionary data
SNARL  : Definitions =  ['to cause to become knotted and intertwined : tangle', 'to make
excessively complicated', 'a tangle especially of hairs or thread : knot', 'a tangled
situation', 'to growl with a snapping, gnashing, or display of teeth', 'to give vent to anger
in surly language', 'a surly angry growl', 'a twisted knot of hairs, thread, etc. : tangle',
'a situation in which you can no longer move or make progress', 'to growl and show the teeth—
usually + at', 'to say something in an angry or annoyed way —often + at', 'an act of growling
and showing the teeth—usually singular']
```

```
SNARL  : Example Sentences =  ['traffic snarls', 'Traffic was snarled up because of the
parade.', 'a traffic snarl-up', 'a brush that smoothes out snarls', 'snarls of string/rope',
'a traffic snarl', 'The project has been plagued by legal/bureaucratic snarls.', 'Her hair
snarls very easily.', 'The rope had become snarled around the post.', 'Get back to work, she
snarled.', 'The stranger snarled at her.']

SNARL  : Words that include the answer =  []


SNARL  : Evaluating definitions
SNARL  : Too similar to the old clue. Passing (to cause to become knotted and intertwined :
tangle)
SNARL  : Evaluating definition (to make excessively complicated)
SNARL  : Similarity is: 18


SNARL  : Too similar to the old clue. Passing (a tangle especially of hairs or thread : knot)
SNARL  : Too similar to the old clue. Passing (a tangled situation)
SNARL  : Evaluating definition (to growl with a snapping, gnashing, or display of teeth)
SNARL  : Similarity is: 46


SNARL  : Evaluating definition (to give vent to anger in surly language)
SNARL  : Similarity is: 35


SNARL  : Evaluating definition (a surly angry growl)
SNARL  : Similarity is: 52


SNARL  : Too similar to the old clue. Passing (a twisted knot of hairs, thread, etc. : tangle)
SNARL  : Evaluating definition (a situation in which you can no longer move or make progress)
SNARL  : Similarity is: 12


SNARL  : Evaluating definition (to growl and show the teeth—usually + at)
SNARL  : Similarity is: 46


SNARL  : Evaluating definition (to say something in an angry or annoyed way —often + at)
SNARL  : Similarity is: 29


SNARL  : Evaluating definition (an act of growling and showing the teeth—usually singular)
SNARL  : Similarity is: 38


SNARL  : Best definition is selected as clue (a surly angry growl)


--------------------------------




DANES  : Getting dictionary data
DANES  : Definitions =  []
DANES  : Example Sentences =  []

DANES  : Words that include the answer =  []


DANES  : Evaluating definitions
DANES  : No definition available to evaluate.
DANES  : Scraping example sentences
DANES  : More Examples = [ The first mentions of Danes are from the 6th century in Jordanes'
Getica, by Procopius, and by Gregory of Tours. , Like the panthers, and lion, and jaguars,
there are huskies, beagles, greyhounds, danes, rottweiler, and a few others. , ...]


DANES  : Evaluating example sentences
DANES  : Evaluating example sentence (Like the panthers, and lion, and jaguars, there are
huskies, beagles, greyhounds, danes, rottweiler, and a few others.)
DANES  : Probability of guessing the word: 0%


DANES  : Scraping Google Knowledge Graph
DANES  : Data from Google Knowledge Graph =  ['Last name of Claire, American actress', 'Last
name of Claire, Claire Catherine _____ is an American actress', 'Album by Neca Falk']


DANES  : Evaluating data
DANES  : Evaluating definition (Last name of Claire, American actress)
DANES  : Similarity is: 25


DANES  : Evaluating definition (Last name of Claire, Claire Catherine _____ is an American
actress)
DANES  : Similarity is: 29


DANES  : Evaluating definition (Album by Neca Falk)
DANES  : Similarity is: 2


DANES  : Best Google Knowledge Graph output is selected as clue (Last name of Claire, American
actress)


---------------------------------




IDEA  : Getting dictionary data
IDEA  : Definitions =  ['a formulated thought or opinion', 'whatever is known or supposed
about something', 'the central meaning or chief end of a particular action or situation', 'a
thought, plan, or suggestion about what to do', 'an opinion or belief', 'something that you
imagine or picture in your mind']

IDEA  : Example Sentences =  ["a child's idea of time", '"Yes we\'re late, but it was raining,
then we got a flat tire …" "All right, I get the idea".', "I think he made a mistake, but
don't get the wrong idea, I still think he has done a good job overall.", "I have no idea what
you're talking about.", "It's a nice idea, but I don't think it'll work.", "I don't have the
faintest/slightest idea what you're talking about.", 'You should put that idea out of your
head.', "I just don't get/understand the idea behind this change in the rules.", 'The whole
idea of the game is to keep from getting caught.', 'The idea is to get people to attend.', 'My
idea is to study law.', 'Starting her own business seemed like a good idea at the time, but it
turned out badly.', 'Whose idea was it to leave so early?', 'My idea was that if we left early
we could beat the crowd.', 'Buying the car was a bad idea.', 'I have some ideas for
redecorating the room.', 'He has an idea for a movie.', "I'm not sure what to do next. Do you
have any ideas?", "She's always full of new ideas.", "It's a good idea to talk to people who
have actually been there.", "What's the next big idea in the fashion industry?", 'Tom has the
right idea —while the rest of us are fighting traffic every day, he takes the train to work.',
'That guy has some pretty strange ideas.', "I thought he'd help us. What gave you that idea?",
'Where did you get that idea?', 'I thought we could handle this ourselves, but my boss had
other ideas. [=my boss did not agree]', 'I formed a good idea of what the place is like by
reading about it.', "A hamburger and a milkshake isn't exactly my idea of a gourmet meal! [=it
is not what I imagine a gourmet meal to be]", 'A quiet night at home is my idea of a good
time.', 'Could you give us some idea of what to expect?', 'He has a clear idea of his
responsibilities. [=he knows what his responsibilities are]', 'Do you have any idea of what
these repairs will cost?', 'Was it hard? You have no idea (how hard it was)! [=yes, it was
very hard]', 'All right, I get the idea. [=I understand]', "I think he made a mistake, but
don't get the wrong idea [=don't misunderstand me], I still think he has done a good job
overall.", "I don't want to give you the wrong idea.", 'The whole idea [=point, object] of the
game is to keep from getting caught.', 'The idea [=goal, aim] is to get people to attend.', "I
just don't get/understand the idea behind [=the reason for] this change in the rules.",
"(informal) Hey! What's the big idea!? [=why are you doing that?]", "He didn't have the
faintest idea/notion what she was talking about. [=he did not know what she was talking
about]", 'He somehow got the idea that I was lying to him.', 'One thing led to another, and—
well, you get the picture/idea. [=you can easily guess the rest]']

IDEA  : Words that include the answer =  []


IDEA  : Evaluating definitions
IDEA  : Evaluating definition (a formulated thought or opinion)
IDEA  : Similarity is: 44


IDEA  : Evaluating definition (whatever is known or supposed about something)
IDEA  : Similarity is: 41


IDEA  : Evaluating definition (the central meaning or chief end of a particular action or
situation)
IDEA  : Similarity is: 23


IDEA  : Evaluating definition (a thought, plan, or suggestion about what to do)
IDEA  : Similarity is: 71


IDEA  : Evaluating definition (an opinion or belief)
IDEA  : Similarity is: 34


IDEA  : Evaluating definition (something that you imagine or picture in your mind)
IDEA  : Similarity is: 51


IDEA  : Best definition is selected as clue (a thought, plan, or suggestion about what to do)
```

```
--------------------------------



PTSD  : Getting dictionary data
PTSD  : Definitions =  ['post-traumatic stress disorder', 'post-traumatic stress disorder']

PTSD  : Example Sentences =  []

PTSD  : Words that include the answer =  []


PTSD  : Evaluating definitions
PTSD  : Word isn't in the dataset. Cannot evaluate definitions.
PTSD  : Scraping example sentences
PTSD  : More Examples =  []


PTSD  : Evaluating example sentences
PTSD  : Scraping Google Knowledge Graph
PTSD  : Data from Google Knowledge Graph =  ['Studio album by Pharoahe Monch', '_____: Post
Traumatic Stress Disorder is the fourth studio album by American hip hop recording artist
Pharoahe Monch, released on April 15, 2014 under his independent label, W', '2014 film']


PTSD  : Evaluating data
PTSD  : Word isn't in the dataset. Cannot evaluate definitions.
PTSD  : A random Google Knowledge Graph output is selected as clue (Studio album by Pharoahe
Monch)


--------------------------------



SINAI  : Getting dictionary data
SINAI  : Definitions =  ['peninsula forming an extension of the continent of Asia in
northeastern Egypt between the Red Sea and the Mediterranean']

SINAI  : Example Sentences =  []

SINAI  : Words that include the answer =  []


SINAI  : Evaluating definitions
SINAI  : Word isn't in the dataset. Cannot evaluate definitions.
SINAI  : Scraping example sentences
SINAI  : More Examples =  []


SINAI  : Evaluating example sentences
SINAI  : Scraping Google Knowledge Graph
SINAI  : Data from Google Knowledge Graph =  ['Cedars-_____ Medical Center, Hospital in Los
Angeles, California', 'Cedars-_____ Medical Center, a non-profit, tertiary 1,400-bed hospital
and multi-specialty academic health science center located in Beverly Grove in the Mid-City
```

```
West area of Los Angeles, California', '_____ Peninsula, Peninsula in Egypt', '_____
Peninsula, ', 'The Mount _____ Hospital, Hospital in New York City, New York', 'Mount _____
Hospital, founded in 1852, is one of the oldest and largest teaching hospitals in the United
States', 'Biblical Mount _____, Mountain', 'Mount _____ is the mountain at which the Ten
Commandments were given to Moses by God, and is one of the most significant of the Stations of
the Exodus', '_____ Health System, Hospital', 'The _____ Health System is a hospital system
which serves Toronto, Ontario, Canada']


SINAI  : Evaluating data
SINAI  : Word isn't in the dataset. Cannot evaluate definitions.
SINAI  : A random Google Knowledge Graph output is selected as clue (Cedars-_____ Medical
Center, Hospital in Los Angeles, California)


--------------------------------




STAND  : Getting dictionary data
STAND  : Definitions =  ['to support oneself on the feet in an erect position', 'to be a
specified height when fully erect', 'to rise to an erect position', 'a halt for defense or
resistance', 'an often defensive effort of some duration or degree of success', 'a stop made
to give a performance', 'to be in an upright position with all of your weight on your feet',
'to move onto your feet from a sitting or low position —often + up', 'to be in an upright
position', 'a strongly held opinion about something—usually singular —often + on', 'a strong
effort to defend yourself or oppose something', 'a partially enclosed structure where things
are sold or displayed']

STAND  : Example Sentences =  ['stand aside', 'can you stand on your head', 'stands accused',
'stands first in the class', 'copy a passage exactly as it stands', 'this book will stand the
test of time', 'stand a siege', 'stand trial', 'stand guard', "I'll stand you a dinner",
'stand drinks', 'stand a stallion', 'a vegetable stand', 'a hot dog stand', 'a taxi stand',
'an ambulance was standing by', 'In December of 1944, it was judged safe to stand down the
Home Guard … after four and a half years of guarding Britain against invasion.', 'Two bowling
pins were left standing.', 'The house she grew up in is no longer standing. [=the house has
been destroyed or knocked down; the house no longer exists]', 'He stands six feet two (inches
tall).', 'The tower stands over 1,000 feet high.', 'Where do we stand financially? [=what is
our financial condition?]', 'She stands accused of murder. [=she has been accused of murder]',
'Where do you stand on the death penalty? Do you think it should be used or not?', "We still
don't know where he stands on this issue.", 'They stand divided [=they disagree] on this
issue.', 'She stands for/against the new regulations. [=she supports/opposes the new
regulations]', 'We ask you to stand (with us) in support of this proposal. [=we ask you to
support this proposal]', 'From where I stand, I think we have to do it.', 'The team still
stands [=ranks] first in the division.', 'She is currently standing in second place.',
'(chiefly Brit) He stands high/low with the voters. [=the voters have a good/bad opinion of
him]', 'The decision still stands.', 'The record she set seems likely to stand for many
years.', 'You must take or leave our offer as it stands.', 'That is how the situation stands
at present.', 'As things stand, we will not be able to meet your deadline.', 'I need a frying
pan that can stand being placed in the oven.', 'These plants can stand [=endure] very cold
temperatures.', 'The team insured their victory with an impressive goal-line stand.', 'The
army is preparing to make a stand against the enemy.', 'students making a stand against the
war', 'a hot-dog/ice-cream/vegetable stand', 'He set up a stand [=booth] at the fair.', 'We
have display stands in many bookstores.', 'concession stands', 'a roadside stand', 'an
umbrella stand', 'a bicycle/microphone stand', 'The ball was hit into the stands.', 'She lied
while on the stand.', 'The witness was asked to take the stand.', 'The magician was booked for
a three-night stand.', 'a stand of pines', 'stand-alone computers', 'stand-alone businesses',
"He was the boss's stand-in during her illness.", 'stand-up comedy', 'a stand-up act/routine',
'a stand-up lunch/bar', 'a stand-up collar', "His friends say he's a real stand-up (kind of)
guy.", 'a stand-up fight/argument', "I've always wanted to try stand-up.", 'a talented stand-
up', "I really hope this wasn't just a one-night stand."]

STAND  : Words that include the answer =  []


STAND  : Evaluating definitions
STAND  : Evaluating definition (to support oneself on the feet in an erect position)
STAND  : Similarity is: 35


STAND  : Evaluating definition (to be a specified height when fully erect)
STAND  : Similarity is: 29


STAND  : Evaluating definition (to rise to an erect position)
STAND  : Similarity is: 37


STAND  : Evaluating definition (a halt for defense or resistance)
STAND  : Similarity is: 16


STAND  : Evaluating definition (an often defensive effort of some duration or degree of
success)
STAND  : Similarity is: 15


STAND  : Evaluating definition (a stop made to give a performance)
STAND  : Similarity is: 32


STAND  : Evaluating definition (to be in an upright position with all of your weight on your
feet)
STAND  : Similarity is: 25


STAND  : Evaluating definition (to move onto your feet from a sitting or low position —often +
up)
STAND  : Similarity is: 32


STAND  : Evaluating definition (to be in an upright position)
STAND  : Similarity is: 26


STAND  : Evaluating definition (a strongly held opinion about something—usually singular —
often + on)
STAND  : Similarity is: 25


STAND  : Evaluating definition (a strong effort to defend yourself or oppose something)
STAND  : Similarity is: 37
STAND  : Evaluating definition (a partially enclosed structure where things are sold or
displayed)
STAND  : Similarity is: 19


STAND  : Best definition is selected as clue (to rise to an erect position)


--------------------------------




THREE  : Getting dictionary data
THREE  : Definitions =  ['a number that is one more than 2', 'the third in a set or series',
'the number 3', 'the third in a set or series']

THREE  : Example Sentences =  ['the three of hearts', 'a three-dimensional analysis of
multiple historical processes', 'three-handed bridge', 'the three of hearts', "What time is
it? It's three.", 'I leave each day at three.', 'a three-cornered hat', 'a three-cornered
agreement', 'a three-dimensional sculpture', 'a three-dimensional image', 'The characters in
the novel are very three-dimensional.', 'three-ply paper', 'three-quarter sleeves']

THREE  : Words that include the answer =  []


THREE  : Evaluating definitions
THREE  : Too similar to the old clue. Passing (a number that is one more than 2)
THREE  : Too similar to the old clue. Passing (the number 3)
THREE  : Scraping example sentences
THREE  : More Examples = [ All matter generally exists in one of three physical phase states
commonly described as solid, liquid, or gas. , I would like to make three brief, concluding
observations by way of answering this question. , ...]


THREE  : Evaluating example sentences
THREE  : Evaluating example sentence (All matter generally exists in one of three physical
phase states commonly described as solid, liquid, or gas.)
THREE  : Probability of guessing the word: 67%


THREE  : Evaluating example sentence (I would like to make three brief, concluding
observations by way of answering this question.)
THREE  : Probability of guessing the word: 0%


THREE  : Evaluating example sentence (The move marks the latest in a string of deals that
could leave the sector in the hands of three or four consolidators.)
THREE  : Probability of guessing the word: 96%


THREE  : Evaluating example sentence (The same applies if a team has a meld of less than seven
pure aces and three or more aces in a player's hand.)
THREE  : Probability of guessing the word: 10%


THREE  : Evaluating example sentence (At the end, if you have completed your canasta of
sevens, each red three you have laid out counts for 100 points bonus.)
```

```
THREE  : Probability of guessing the word: 0%


THREE  : Evaluating example sentence (Declare a gang and gun amnesty to be followed by a three
to six-month period of detailed observation and prosecution.)
THREE  : Probability of guessing the word: 27%


THREE  : Evaluating example sentence (Business start-ups in the borough are up by 29 per cent
for the first three months of the year.)
THREE  : Probability of guessing the word: 10%


THREE  : Evaluating example sentence (In the space of three hours we were told help should be
focused on start-ups or businesses with real growth potential.)
THREE  : Probability of guessing the word: 11%


THREE  : Evaluating example sentence (At noon today a three minute silence will be observed
across Europe in remembrance of the victims of the Sumatran tsunami.)
THREE  : Probability of guessing the word: 3%


THREE  : Evaluating example sentence (At school, our classroom had a small rodent zoo
consisting of two rabbits, three hamsters, a litter of baby gerbils and a guinea pig.)
THREE  : Probability of guessing the word: 9%


THREE  : Evaluating example sentence (Twenty three dead animals were removed from the
premises, some of them had cannibalised each other.)
THREE  : Probability of guessing the word: 7%


THREE  : Evaluating example sentence (I have already seen it three times and each time I gain
new insights into my own state of mind.)
THREE  : Probability of guessing the word: 4%


THREE  : Evaluating example sentence (Water is the only common substance that occurs naturally
on earth in three different physical states.)
THREE  : Probability of guessing the word: 11%


THREE  : Evaluating example sentence (Cannibalism was one method of establishing otherness in
early modern representations of all three groups.)
THREE  : Probability of guessing the word: 0%


THREE  : Evaluating example sentence (As what seemed to be the result of the flash, the three
hands of the clock began to speed up.)
THREE  : Probability of guessing the word: 1%


THREE  : Evaluating example sentence (Simply, Le Guen believes every coach has a shelf-life of
three or four years at any one club before he grows stale and people turn against him.)
THREE  : Probability of guessing the word: 99%
THREE  : Most guessable example sentence is selected as clue (Simply, Le Guen believes every
coach has a shelf-life of _____ or four years at any one club before he grows stale and people
turn against him.)


---------------------------------




ELSA  : Getting dictionary data
ELSA  : Definitions =  []

ELSA  : Example Sentences =  []

ELSA  : Words that include the answer =  []


ELSA  : Evaluating definitions
ELSA  : No definition available to evaluate.
ELSA  : Scraping example sentences
ELSA  : More Examples =  []


ELSA  : Evaluating example sentences
ELSA  : Scraping Google Knowledge Graph
ELSA  : Data from Google Knowledge Graph =  ['Fictional character', "_____ of Arendelle is a
fictional character who appears in Walt Disney Animation Studios' 53rd animated film Frozen
and its sequel Frozen II", 'First name of Hosk, Swedish model', "First name of Hosk, _____
Anna Sofie Hosk is a Swedish model and current Victoria's Secret Angel, who has worked for
brands including Dior, Dolce & Gabbana, Free People, Ungaro, H&M, Anna Sui, Lilly Pulitzer and
Guess", 'First name of Pataky, Spanish model', 'First name of Pataky, ', 'River in Italy',
'First name of Peretti, Italian jewelry designer', 'First name of Peretti, an Italian jewelry
designer and philanthropist as well as a former fashion model', 'First name of Martinelli,
Italian actress', 'First name of Martinelli, _____ Martinelli was an Italian actress and
fashion model']


ELSA  : Evaluating data
ELSA  : Word isn't in the dataset. Cannot evaluate definitions.
ELSA  : A random Google Knowledge Graph output is selected as clue (Fictional character)


---------------------------------




PSST :
       Old clue: "Hey, you! Over here!"
       New clue: —used to get someone's attention


TITHE :
       Old clue: Give 10% to the church
       New clue: a tenth part of something paid as a voluntary contribution or as a tax
especially for the support of a religious establishment
```

```
SNARL :
       Old clue: Hair tangle
       New clue: a surly angry growl


DANES :
       Old clue: "Great" dogs
       New clue: Last name of Claire, American actress


IDEA :
       Old clue: Inkling
       New clue: a thought, plan, or suggestion about what to do


PTSD :
       Old clue: Condition for a returning combat vet
       New clue: Studio album by Pharoahe Monch


SINAI :
       Old clue: Mount where Moses received the Ten Commandments
       New clue: Cedars-_____ Medical Center, Hospital in Los Angeles, California


STAND :
       Old clue: Roadside produce seller
       New clue: to rise to an erect position


THREE :
       Old clue: Number of bones in the human ear
       New clue: Simply, Le Guen believes every coach has a shelf-life of _____ or four years
at any one club before he grows stale and people turn against him.


ELSA :
       Old clue: Queen in "Frozen 2"
       New clue: Fictional character


Clue generation completed.
```