CS 461-Artificial Intelligence

Term Project Report

POWERPUZZ

Bayram Muradov **21503664**
Skerd Xhafa **21503353**

SECTION: 1

DATE: 27.12.2019

Table of Contents

1.0 Introduction

Puzzles are designed in order to test one's understanding of a particular topic or evaluate person's general knowledge in various mediums. Additionally, crossword puzzles are good tools for enhancing the level of learning of students. Edward K. Crossman and Sharyn M. Crossman state that, using crossword puzzles can be beneficial in courses that require memorization of some important relationships [1]. Today, various newspapers design their own puzzles in order to indulge the puzzle lovers. Puzzle audiences come from different backgrounds with various ages. Process of generating puzzles is therefore hard since the puzzle designer needs to address all these different people. Probably this hard and time taking process is the main reason why there's only one puzzle generated for each day. In order to ease the burden of puzzle design, our project aims to generate new set of clues given the current ones and the solution of the puzzle. In order to meet course requirements, our team was assigned to generate alternative clues to the puzzle designed by Joel Fagliano for New York Times newspaper [2]. Having different clues on the same puzzle would widen the general knowledge of the puzzle solver even more. In order to provide our users with meaningful set of clues our program obtains the puzzle of the day by downloading it from the puzzle website and creates the local version of the same puzzle by preserving the official solutions and generating alternative clues for the same set of solutions. Our team hopes that puzzle lovers will get most out of the program.

2.0 Problem Approach

Our team started to research about the potential ways to solve the problems related to crossword puzzles immediately after the assignment of the project. Through our research we have encountered various interesting methodologies in order to solve the problems related to crossword puzzles including finding the number of solutions to a given crossword puzzle by using Bayesian Estimation by G.H. Harris and J.J.H. Forster [3]. However, since we already have taken the puzzle geometry and solutions as input, finding number of solutions to the puzzle was out of the scope of our project. After exploring mathematical approaches to solve the problem and deciding they would not be useful we decided to take a semantic approach in order to solve the problem. David E. Goldschmidt and Mukkai Krishnamoorthy took a semantic approach and implemented Google CruciVerbalist (GCV) to solve the crossword puzzles by reformulating the clues [4]. In order to move to the semantic approach, we have started to search for dictionaries that could be used in alternative clue generation process. After inputting solutions to these dictionaries we were getting definitions of the words that for the most of the time were different from the ones that were not available in the official set of

clues. The main source of dictionaries in our programs were the ones that have provided a web API for our program to use. After making some observation we have decided to use three properties of the dictionary outputs to form the new set of clues including definition, synonym and a sentence in which a solution word has been used. Definitions were mostly used to describe the word which we found perfectly suitable for the clue generation procedure. However, in the dictionaries we used, sometimes the solution to the clue has also be shown in the definition of a particular word. In order to solve this issue, we have filtered out the word from the output of dictionary in case of its inclusion. For example, if the sentence is 'Elliot is the main character in Mr. Robot' and the solution to the clue is 'Elliot', we filter out 'Elliot' from the sentence leaving '____ is the main character in Mr. Robot' to the generated set of clues. Synonym of the word can be used in a pretty effective manner as well in case the word has various synonyms. Lastly, using the sentence in clue generation is a widely used methodology in practice of clue generation as our observations have shown. However, since the sentences related with a particular word would actually contain that same word, additional measures have been taken in order to filter out the solution from the example sentence like filtering out the word that contains the solution to the clue. In case of getting no result for a particular word (mostly the acronyms), we use our custom clue generators like displays the word in reverse order. Overall, by combining all of these strategies we get strong method to form the set of clues that are alternative to the original ones.

3.0 Resources

In order to solve the problem we have searched though different dictionaries which are the main sources that have been used for the project. First dictionary that have been used for the purposes of the project is popular english dictionary called Oxford [5]. This dictionary is very useful in order to find the different definitions of the word. Queries to Oxford dictionary returns with a result that can be used in clue generation procedure. However, in case there is no result from Oxford dictionary or API has failed and is out of service for some reason, Urban Dictionary has been used in order to provide results [6]. Since probability of getting jargon or offensive word is high while using urban dictionary, additional measures have been taken to check if the output of the dictionary includes a profanity. In case of getting profanity, another dictionary API called DataMuse has been used in order to provide results [7]. DataMuse gives a set of synonyms for the entered input which is used as a back up plan in case the dictionaries above fail. In case of not finding the desired results in the dictionaries given above, custom methods have been implemented in order to generate the set of alternative clues like asking for the reverse of the word. All the outputs from the

dictionaries have been randomized and users get different set of clues each time they run the program which we think is beneficial for the users who wish to learn new information while solving the puzzles.

4.0 Implementation Details

Implementation details of the project including used algorithms, data structures and followed procedure to generate alternative set of clues have been discussed in detail in this section of the document together with the trade-offs that have been obtained from the implementation choices.

4.1 Algorithms Used

Since the strategy we followed was to search through some number of dictionaries in order to generate alternative clues, we have used iterative searching in order to obtain meaningful set of results. Our iterative searching algorithm has order of search priority while searching dictionaries since it doesn't search the next dictionary in case of obtaining the desired output which we consider as significant improvement in terms of efficiency. First dictionary we search is popular English dictionary called Oxford. In case of not getting the desired output, Urban Dictionary has been consulted. Outputs from this dictionary have been checked for profanity and in case of inclusion of offensive word, output was considered as unsuccessful and DataMuse dictionary has been inputted with the desired word. In case of getting desired output set, outputs have been randomized and have been fed to the output set.

4.2 Data Structures Used

In order to keep the puzzle data from the website ArrayList collection of Java has been used. Instead of reading the HTML contents of the website, writing them to a file and reading from the same file, using local data structure was more efficient choice since file input/output operations may take significant amount of time. In order to provide synchronization between the methods that access the globals, timer based semaphore implementations of Java Selenium library have been used [8]. 5x5 matrices have been used in order to keep the geometry of the puzzle. HashMaps have been used in order to keep clues with their related solution keys.

4.3 Clue Generation Process

After obtaining the puzzle data including puzzle geometry, the set of current clues, and official solutions clue generation process. We have isolated this core part of the project from puzzle download and GUI creation part in order to achieve modularity and good level of organization in the project structure. With the

separation of clue generation process from the puzzle download and GUI, only an instance of ClueGenerator method has been used together with its 'getClues' method in order to obtain new set of clues. Sources used to get clues have been discussed in detail below.

- **Oxford Dictionary**

In order to search the Oxford Dictionary from the code, we needed an access to their dictionary API. In order to use the API, some credentials were needed to be obtained from the official web site of Oxford dictionary. 'Prototype' version was chosen from the available 3 packages since it was free to use until 1000 queries per month which were enough in order to test the program. After getting the credentials, calls were made from the code and response got from the Oxford API. Since the response was in JSON format, with different word properties included in them, some parsing was done in order to filter out the needed words from the response. Custom parsing methods have been implemented after observing the content of the JSON data. Results obtained from Oxford Dictionary were mostly satisfying the requirements of our clue generation strategy. However, sometimes API was failing to service when the number of queries exceeded for some particular period of time (2 minutes usually). The main reason for that was the usage of prototype package as stated earlier. In case of failure, other dictionaries have been consulted.

- **Urban Dictionary**

Urban Dictionary has been chosen as a back up plan in case the Oxford dictionary fails for the reasons stated above or returns no output. The words in urban dictionary are being defined by regular users of the website and since it gives output for every inputted word. However, since common language also may include some profanity, results got from this dictionary have been checked against the list of 'bad words'. Urban Dictionary provides an open end, web API that doesn't require any credentials to be accessed. After inputting query to the API, it returns an output in JSON format. Outputs have been parsed from the JSON in order to get meaningful results by using the parsing functions we have implemented. Using the dictionary was free of charge and didn't require to obtain any credentials.

- **DataMuse**

DataMuse is an API designed for developers to identify the context of the word or sentence and output a list of words with similar context. It is an open-end, free to use

tool which doesn't require any authentication. Output of the API is JSON array that contains the list of words that are similar to the inputted one. In order to get the actual words inside of the JSON array, custom parser functions have been used.

The results got from this API were mostly the synonyms of the solution words and sometimes were hard to be used as a clue. However, since this API is used in the third layer of clue generation process, it is not being used very often by the program.

After designing the different functions related to the previously stated dictionaries we were able to, search for the alternative set of clues. As stated earlier, our methods use random function in order to generate different set of clues at each run. We have implemented functions in highly modular manner that is, our main clue generator function takes the solution word as input and calls the helper functions which have the same prototype and are responsible for searching for the clues in three different APIs stated previously. Our clue generation procedure is as follows: we are iterating the API outputs in order to find the clues that meet the requirements of certain constraints. First constraint is that the outputs can not contain any profanity in them. This constrained has been taken into consideration in order to filter out the jargons or offensive words produced by Urban Dictionary. Another constraint is that the outputs cannot contain any word that are same with the official solution. We have implemented some additional methods in order to filter those words out and replace them with '_____' symbol. This case appears mostly in the outputs that give a sentence for the inputted word. Third constraint is that, the word inputted cannot be same with the outputs got from the APIs. This situation occurs mostly when using the DataMuse API which is the last layer of clue search and considered as a failure. If a failure occurs in the last layer of the clue search procedure, custom method that reverses the clue becomes operational. For example, if the input is 'TFA' and no results have been obtained from the dictionary searches, 'RFA in reverse' or 'RFA in mirror' will be added to generated clue set depending on the output of the Java's random algorithm. This function is mostly being called for the acronyms in the official solution set for which dictionaries return no result that satisfy the mentioned constraints.

Finally, a method has been implemented in order to call the clue generator and input puzzle data into it. This method takes the puzzle data as an input and returns the newly generated classified into across and down categories. After the implementation of this particular interface, function has been called from the GUI part of the project.

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

(Word Count: 2260)

References

[1] - The Crossword Puzzle as a Teaching Tool, Edward K. Crossman and Sharyn M. Crossman.

[2] - NYTimes, "NYTimes Crossword," 25 december 2019. [Online]. Available: https://www.nytimes.com/crosswords/game/mini.

[3] - On the Bayesian Estimation and Computation of the Number of Solutions to Crossword Puzzles, G.H. Harris and J.J.H. Forster.

[4] - Comparing keyword search to semantic search: a case study in solving crossword puzzles using the Google™ API, David E. Goldschmidt and Mukkai Krishnamoorthy

[5] - Oxford, "Oxford Dictionaries," Oxford Dictionary, [Online]. Available: https://developer.oxforddictionaries.com/. [Accessed 25 december 2019].

[6] - U. Dictionary, "Rapid API," Rapid API, [Online]. Available: https://rapidapi.com/community/api/urban-dictionary. [Accessed 25 december 2019].

[7] - DataMuse API, "DataMuse" [Online]. Available: https://www.datamuse.com/api/. [Accessed 25 december 2019].

[8] - "Selenium with Java," Selenium, [Online]. Available: https://selenium.dev/documentation/en/. [Accessed 25 december 2019].

Appendix

Code Segments:

Clue generation related codes:

Class below searches the three dictionaries to generate alternative set of clues:

```
package aiproject;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.ArrayList;

public class ClueGenerator {

    //properties
    public ArrayList<String[]> currentClues;
    public ArrayList<String[]> newClues;
    public ArrayList<String> profanityList;
    public ArrayList<Object> newDownClues;
    public ArrayList<Object> newAccrossClues;
    //dictionaries
    Oxford oxfordDict;
    UrbanDictionary urbanDictionary;
    DataMuse dataMuse;

    //constructor
    public ClueGenerator(ArrayList<String[]> clues){
        currentClues = clues;
        newClues=new ArrayList<String[]>();
        newDownClues = new ArrayList<Object>();
        newAccrossClues = new ArrayList<Object>();
        try {
            oxfordDict=new Oxford();
```

```
        }
        catch (Exception e) {
            e.printStackTrace();
        }
        urbanDictionary=new UrbanDictionary();
        dataMuse=new DataMuse();
        profanityList=new ArrayList<>();
        profanityList.add(PROFANITY_WORDS);
    }
    public ClueGenerator(){
        try {
            oxfordDict=new Oxford();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
        urbanDictionary=new UrbanDictionary();
        dataMuse=new DataMuse();
    }

    //methods
    public void printCurrClues(){
        for (int i=0; i<currentClues.size(); i++) {
            System.out.println(currentClues.get(i));
        }
    }

    public void printNewClues() {
        for(int i=0; i<newClues.size(); i++) {
            System.out.println(newClues.get(i)[2]);
        }
    }
```

```
//**CLUE GENERATOR methods**

//generates new clues for inputted set of current clues
  public void generateClues() {
      for(int i=0; i<currentClues.size(); i++) {
          String curr= currentClues.get(i)[2];
          //1st: check OXFORD

              String newClue = "no result";
              if(checkProfanity(getOxfordClue(curr))){
                newClue = getOxfordClue(curr);
                if(checkIncluded(newClue, curr)) {
                  replaceBlank( curr, newClue);
                }
              }
              //2nd: check UrbanDict'
              if(newClue.equals("no result") ) {
                if(checkProfanity(getUrbanClue(curr)))
                  newClue = getUrbanClue(curr);
                  if(checkIncluded(newClue, curr)) {
                    replaceBlank( curr, newClue);
                  }
              }
              //3rd: check DataMuse
              if (newClue.equals("no result")) {
                  if(checkProfanity(getDataMuseClue(curr))) {
                      newClue = getDataMuseClue(curr); }
                      if(checkIncluded(newClue, curr)) {
                        replaceBlank( curr, newClue);
                      }
                  }
              }
              if (newClue.equals("no result")) {
```

```
      //check the size; if acronym reverse else keep original
                        if(curr.length<3) {
                          newClue=reverseGen(curr);
                          System.out.println("**reversing the
original clue**");
                        }
                        else{
                          //if nothing found, keep the original
clue
                          System.out.println("**keeping the
original clue**");
                          newClue = curr;
                        }
                    }
                    ArrayList<Object> temp = new
ArrayList<Object>();
                    temp.add(currentClues.get(i)[0]);
                    temp.add(newClue);

                    if( currentClues.get(i)[1].equals("Down") ){
                        newDownClues.add(temp);
                    }
                    else{
                        newAccrossClues.add(temp);
                    }

                    newClues.add( new String[]
{ currentClues.get(i)[0], currentClues.get(i)[1], newClue } );

                    System.out.println("The received clue is:\n
[" + newClue + "]");

                }

            }
```

```
      }

    //gets clue from DataMuse API
    public String getDataMuseClue(String curr) {
        System.out.println("**getting clues from DataMuse
dictionary**[" + curr + "]");
        String newClue= dataMuse.findSimilar(curr);
        if(newClue.equals("no result")) {
            System.out.println("**no result from DataMuse
dictionary**");
        }
        return newClue;
    }

    //gets clue from UrbanDict API
    public String getUrbanClue(String curr) {
//        System.out.println("inside urban: "+curr);
        System.out.println("**getting clues from
UrbanDictionary**[" + curr + "]");
        String newClue= urbanDictionary.findSimilar(curr);
       // System.out.println("newClue in urban: "+ newClue);
        if(newClue.equals("no result")) {
            System.out.println("**no result from
UrbanDictionary**");
        }
        return newClue;
    }


    //gets clue from Oxford API
    public String getOxfordClue(String curr) {
        System.out.println("**getting clues from OXFORD
dictionary** [" + curr + "]");
        String newClue= oxfordDict.dictionary(curr);
```

```
        if(newClue.equals("no result")) {
            System.out.println("**no result from OXFORD
dictionary**");
        }
        return newClue;
    }

    //checks for profanity in the clue candidate
    public boolean checkProfanity(String word) {
      for(int i=0; i<profanityList.size(); i++) {
        if(profanityList.get(i).equals(word)){
          return false;
        }
      }
      return true;
    }

    //reverses the String: custom clue generator
    public String reverseGen(String word) {
        ArrayList <String> newClues=new ArrayList<>();
        newClues.add("in reverse");
        newClues.add("in the mirror")
        String reverse = "";
        for(int i = str.length() - 1; i >= 0; i--)
        {
            reverse = reverse + word.charAt(i);
        }
        int rnd = (int) Math.random() * newClues.size();
        String result= reverse+newClues.get(rnd);
        return result;
    }

    //checks if the word is included in newclue
```

```
    public boolean checkIncluded(String word, String newClue)
{
      if(newClue.contains(word)) {
        return true;
      }
      return false;
    }

    //replaces the current word  in newClue with blank
    public String replaceBlank(String curr, String word) {
      return word.replaceAll(curr , "____");
    }

}
```

Class below searches DataMuse API to find clues:

```
package aiproject;
import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLConnection;
import java.util.ArrayList;

/**
 * Created by bmmuradov on 19.12.2019.
 */
public class DataMuse {
```

```
    //properties

    //constructor
    public DataMuse() {
    }

    public String findSimilar(String word) {
        String s = word.replaceAll(" ", "+");
        return getWord(getJSON("http://api.datamuse.com/words?
rd="+s));
    }

    private String getJSON(String url) {
        URL datamuse;
        URLConnection dc;
        StringBuilder s = null;
        try {
            datamuse = new URL(url);
            dc = datamuse.openConnection();
            BufferedReader in = new BufferedReader(new
InputStreamReader(dc.getInputStream(), "UTF-8"));
            String inputLine;
            s = new StringBuilder();
            while ((inputLine = in.readLine()) != null)
                s.append(inputLine);
            in.close();
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return s != null ? s.toString() : null;
    }
```

```
    //parses the json String to get word data
    private String getWord(String jsonStr) {
        ArrayList<String> defs= new ArrayList<>();
        String result=null;
        JSONParser parser= new JSONParser();
        Object parse=null;
        try{
            parse=parser.parse(jsonStr);
        }
        catch (Exception e) {
            e.printStackTrace();
        }
        JSONArray jsonArray=(JSONArray) parse;
        for (Object e:jsonArray) {
            JSONObject senJson= (JSONObject) e;
            result=senJson.get("word").toString();
            result=shorthen(result);
            defs.add(result);
        }
        if(defs.size()==0) {
            //System.out.println("no result from DataMuse");
            return "no result";
        }
        else {
            //randomize result
            int rnd = (int) Math.random() * defs.size();
            result = defs.get(rnd);
            return result;
        }
    }
    //gets the srting util dull stop in case the string is
long
    private String shorten(String word) {
      int iend = word.indexOf(".");
```

```
      String subString =word;
      if (iend != -1)
      {
          subString= filename.substring(0 , iend);
      }
      return substring

    }
}
```

Class below searches Oxford API to find clues:

```
package aiproject;

/**
 * Created by bmmuradov on 19.12.2019.
 */
import  org.json.simple.JSONArray;
import  org.json.simple.JSONObject;
import  org.json.simple.parser.JSONParser;
import  javax.net.ssl.HttpsURLConnection;
import java.io.BufferedReader;
import  java.io.InputStreamReader;
import  java.net.URL;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
import java.util.ArrayList;

public class Oxford extends UnicastRemoteObject {


    public Oxford() throws RemoteException {
    }
```

```
    public String dictionary(String word) {
        ArrayList<String> defs=new ArrayList<>();
        //System.out.println("**inside dictionary**");
        JSONParser parser = new JSONParser();
        String result="";
        Object parse=null;
        try {
            result = getRequest(buildURL(word));
            parse = parser.parse(result);
            JSONObject jsonObject=(JSONObject) parse;
            JSONArray res=(JSONArray)
jsonObject.get("results");
            for (Object e:res){
                JSONArray lexicalEntries= (JSONArray)
(((JSONObject) e).get("lexicalEntries"));
                for(Object lex:lexicalEntries) {
                    JSONArray entries= (JSONArray)
(((JSONObject) lex).get("entries"));
                    for(Object ent:entries) {
                        JSONArray senses= (JSONArray)
(((JSONObject) ent).get("senses"));
                        for(Object sen:senses) {
                            JSONObject senJson= (JSONObject)
sen;

result=senJson.get("definitions").toString();
                            result=result.replaceAll("\\[",
"").replaceAll("\\]","");
                            result=result.replaceAll("\"",
"");
                            result=shorten(result);
                            defs.add(result);
                        }
                    }
                }
```

```
            }

        } catch (Exception e) {
            //e.printStackTrace();
            return "no result";
        }
        if(defs.size()==0) {
            //System.out.println("no result from
OxfordDictionary");
            return "no result";
        }
        else {
            //randomize result
            int rnd = (int) Math.random() * defs.size();
            result = defs.get(rnd);
            return result;
        }
    }

    private String buildURL(final String word) {
        final String language = "en";
        final String word_id = word.toLowerCase();
        return "https://od-api.oxforddictionaries.com:443/api/
v2/entries/" + language + "/" + word_id;
    }


    private String getRequest(String link) {
        final String app_id = "";
        final String app_key = "";
        try {
            URL url = new URL(link);
            HttpsURLConnection urlConnection =
(HttpsURLConnection) url.openConnection();
```

```
            urlConnection.setRequestProperty("Accept",
"application/json");
            urlConnection.setRequestProperty("app_id",
app_id);
            urlConnection.setRequestProperty("app_key",
app_key);
            // read the output from the server
            BufferedReader reader = new BufferedReader(new
InputStreamReader(urlConnection.getInputStream()));
            StringBuilder stringBuilder = new StringBuilder();
            String line = null;
            while ((line = reader.readLine()) != null) {
                stringBuilder.append(line + "\n");
            }
            return stringBuilder.toString();
        } catch (Exception e) {
            e.printStackTrace();
            return e.toString();
        }
    }
    //gets the srting util dull stop in case the string is
long
    private String shorten(String word) {
      int iend = word.indexOf(".");
      String subString =word;
      if (iend != -1)
      {
          subString= filename.substring(0 , iend);
      }
      return substring

    }
}
```

Class below searches UrbanDictionary API to find clues:

```
package aiproject;
import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import  org.json.simple.parser.JSONParser;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLConnection;
import java.util.ArrayList;

/**
 * Created by bmmuradov on 20.12.2019.
 */
public class UrbanDictionary {

    //properties

    //constructors
    public UrbanDictionary(){
    }


    public String findSimilar(String word) {
        String s = word.replaceAll(" ", "+");
        //System.out.println("inside findSim: "+s);
        return getWord(getJSON("http://
api.urbandictionary.com/v0/define?term="+s));
    }

    private String getJSON(String url) {
        URL datamuse;
```

```
        URLConnection dc;
        StringBuilder s = null;
        try {
            datamuse = new URL(url);
            dc = datamuse.openConnection();
            BufferedReader in = new BufferedReader(new
InputStreamReader(dc.getInputStream(), "UTF-8"));
            String inputLine;
            s = new StringBuilder();
            while ((inputLine = in.readLine()) != null)
                s.append(inputLine);
            in.close();
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return s != null ? s.toString() : null;
    }

    //parses the json String to get word data
    private String getWord(String jsonStr) {
        ArrayList<String> defs= new ArrayList<>();
        String result=null;
        JSONParser parser= new JSONParser();
        Object parse=null;
        try{
            parse=parser.parse(jsonStr);
        }
        catch (Exception e) {
            e.printStackTrace();
        }
        JSONObject jsonObject=(JSONObject) parse;
        //System.out.println(jsonObject.toString());
```

```
        JSONArray res=(JSONArray) jsonObject.get("list");

        for(Object e: res) {
            JSONObject senJson= (JSONObject) e;
            result=senJson.get("definition").toString();
            result = result.replaceAll("\\[",
"").replaceAll("\\]","");
            result= shorten(result);
            defs.add(result);
        }
        if(defs.size()==0) {
            //System.out.println("no result from
UrbanDictionary");
            return "no result";
        }
        else {
            //randomize result
            int rnd = (int) Math.random() * defs.size();
            result = defs.get(rnd);
            return result;
        }
    }
    //gets the srting util dull stop in case the string is
long
    private String shorten(String word) {
      int iend = word.indexOf(".");
      String subString =word;
      if (iend != -1)
      {
          subString= filename.substring(0 , iend);
      }
      return substring

    }
```

```
}
```

Puzzle download related codes:

Class below downloads puzzle data including puzzle geometry, clues and official solution:

```
package aiproject;

/**
 * Created by bmmuradov on 4.11.2019.
 */

import UI.LoadingPanel;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;
import org.openqa.selenium.By;
import org.openqa.selenium.TimeoutException;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

import java.util.ArrayList;
import java.util.List;
import org.openqa.selenium.WebElement;

public class PuzzleGeometry {

    private WebDriver webDriver;
    public int[][] puzzleMatrix;
```

```
    public String[][] puzzleMatrixString;
    public String[][] puzzleMatrixQuestionNumbers;

    public static ArrayList<Object> allTiles ;


    //timeout until the page loads fully
    private static final int TIMEOUT=30;

    public PuzzleGeometry(){

        //System.setProperty("webdriver.chrome.driver","C:\
\Users\\skerd\\Documents\\NetBeansProjects\\Cs461\\chrom\
\chromedriver.exe");
        System.setProperty("webdriver.chrome.driver","C:\
\Users\\skerd\\Documents\\NetBeansProjects\\Cs461\\chrom\
\chromedriver.exe");
        System.setProperty("webdriver.gecko.driver", "./
geckodriver");
        ChromeOptions options = new ChromeOptions();
        webDriver = new ChromeDriver(options);

        allTiles = new ArrayList<Object>();

        try {
            //open puzzle page
            webDriver.get("https://www.nytimes.com/crosswords/
game/mini");

            WebDriverWait wait = new WebDriverWait(webDriver,
TIMEOUT);
```

```
            //
================================================================
======

            List<WebElement> links =
webDriver.findElements(By.className("ClueList-wrapper--3m-
kd"));

            List<WebElement> across =
links.get(0).findElements(By.className("Clue-li--1JoPu"));
            int numberOfAcrossClues = across.size();

            List<WebElement> down =
links.get(1).findElements(By.className("Clue-li--1JoPu"));
            int numberOfDownClues = down.size();

            System.out.println("Number of across clues: " +
numberOfAcrossClues);
            System.out.println("Number of down clues: " +
numberOfDownClues);

            ArrayList<Object> acrossClues = new
ArrayList<Object>();
            ArrayList<Object> downClues = new
ArrayList<Object>();

            for(WebElement we: across) {

                String c = we.getText().replace("\n", " ");

                int questionNr =
Integer.parseInt( c.substring(0, c.indexOf(" "))  );
                String question = c.substring( c.indexOf(" ")
+1 );
                System.out.println("[A][" + questionNr + "]["
+ question + "]");
```

```
                ArrayList<Object> temp = new
ArrayList<Object>();
                temp.add(questionNr);
                temp.add(question);

                acrossClues.add(temp);

            }

            for(WebElement we: down) {
                    String c = we.getText().replace("\n", "
");

                    int questionNr =
Integer.parseInt( c.substring(0, c.indexOf(" "))  );
                    String question = c.substring( c.indexOf("
") +1 );

                    System.out.println("[D][" + questionNr +
"][" + question + "]");

                    ArrayList<Object> temp = new
ArrayList<Object>();
                    temp.add(questionNr);
                    temp.add(question);

                    downClues.add(temp);

            }


            //
==========================================================
======
```

```
            //wait until start button appears

wait.until(ExpectedConditions.visibilityOfElementLocated(By.cl
assName("buttons-modalButton--1REsR")));

            System.out.println("clicking start");
            //click START
            webDriver.findElements(By.className("buttons-
modalButton--1REsR")).get(0).click();
            System.out.println("start clicked");

            //wait until toolbar/REVEAL appears

wait.until(ExpectedConditions.visibilityOfElementLocated(By.xp
ath("//*[@id=\"root\"]/div/div[1]/div[4]/div/main/div[2]/div/
div/ul/div[2]/li[2]/button")));

            //click REVEAL
            webDriver.findElement(By.xpath("//*[@id=\"root\"]/
div/div[1]/div[4]/div/main/div[2]/div/div/ul/div[2]/li[2]/
button")).click();

            //click PUZZLE
            webDriver.findElement(By.xpath("//*[@id=\"root\"]/
div/div[1]/div[4]/div/main/div[2]/div/div/ul/div[2]/li[2]/ul/
li[3]/a")).click();

            //wait until pop-up/REVEAL appears

wait.until(ExpectedConditions.visibilityOfElementLocated(By.xp
ath("//*[@id=\"root\"]/div/div[2]/div[2]/article/div[2]/
button[2]")));
```

```
            //click REVEAL on pop-up
            webDriver.findElement(By.xpath("//*[@id=\"root\"]/
div/div[2]/div[2]/article/div[2]/button[2]")).click();

            //get page source
            String pageSource = webDriver.getPageSource();

            //parse page source
            Document doc = Jsoup.parse(pageSource);
            Elements groups = doc.select("#xwd-board > g:nth-
child(5)");
            //get 'g' tags
            Elements cells = groups.first().children();

            int tracer=0;
            puzzleMatrix = new int[5][5];
            puzzleMatrixString = new String[5][5];
            puzzleMatrixQuestionNumbers = new String[5][5];
            int i=0;
            int j=0;

            for (Element e : cells){
                if(i==5) {
                    break;
                }
                if(j==5) {
                    j=0;
                    i++;
                }
                puzzleMatrix[i][j] = processCellData(e, i, j);
                j++;
            }
```

```
            printMatrixWithSolution();



            ClueGenerator newGen = new
ClueGenerator(allSolutionInformation());
            newGen.generateClues();
            newGen.printNewClues();

            LoadingPanel.newAccrossClues =
newGen.newAccrossClues;
            LoadingPanel.newDownClues = newGen.newDownClues;
            LoadingPanel.aaClues = acrossClues;
            LoadingPanel.ddClues = downClues;
            LoadingPanel.tiles = allTiles;
            LoadingPanel.haveAllQuestions = true;

            webDriver.close();
            webDriver.quit();
        }
        catch (TimeoutException e) {
            System.out.println("timed out");
            System.exit(0);
        }

    }

    private int processCellData(Element element, int row, int
col){
        Elements components =
element.getElementsByTag("text");
        int elSize= components.size();
        int result=-1;
        //3 cases
        //1. number& letter: edge cells
```

```
        //2. only letters: inner cells
        //3. nothing: black cells

        if(elSize==4) { //number&letter
            String number= components.get(0).text();
            String letter= components.get(3).text();

            //System.out.println("N: [" + number + "]  L[" +
letter + "]");

            int newNumber = -1;

            if( number != null){
               newNumber = Integer.parseInt(number);
            }

            ArrayList<Object> temp = new ArrayList<Object>();
            temp.add(newNumber);  temp.add(""); temp.add(row);
temp.add(col);
            temp.add(true); temp.add(letter);
            puzzleMatrixString[row][col] = letter;
            puzzleMatrixQuestionNumbers[row][col] = number;

            //System.out.println("Adding number+letter: [" +
number + "][" + letter + "][" + row  +"][" + col + "]");

            allTiles.add(temp);

            result=2;
        }
        else if(elSize==2) { //only letter
            String letter= components.get(1).text();
            String number = null;
```

```
            int newNumber = -1;

            if( number != null){
                newNumber = Integer.parseInt(number);
            }

            ArrayList<Object> temp = new ArrayList<Object>();
            temp.add(newNumber);  temp.add(""); temp.add(row);
temp.add(col);
            temp.add(true); temp.add(letter);
            puzzleMatrixString[row][col] = letter;
            puzzleMatrixQuestionNumbers[row][col] = number;

            //System.out.println("Adding only letter: [" +
number + "][" + letter + "][" + row  +"][" + col + "]");

            allTiles.add(temp);

            result=1;
        }
        else if(elSize==0) {//black cell
            String letter=null;
            String number=null;

            int newNumber = -1;

            if( number != null){
                newNumber = Integer.parseInt(number);
            }

            //System.out.println("Adding Black cell: [" +
number + "][" + letter + "][" + row  +"][" + col + "]");

            ArrayList<Object> temp = new ArrayList<Object>();
```

```
            temp.add(newNumber);  temp.add(""); temp.add(row);
temp.add(col);
            temp.add(false); temp.add(letter);
            puzzleMatrixString[row][col] = "~";
            puzzleMatrixQuestionNumbers[row][col] = "B";

            allTiles.add(temp);

            result=0;
        }

       return result;

    }


    private void printMatrixWithSolution(){

        for( int i = 0; i < 5; i++){
            for( int j = 0; j < 5; j++){
                System.out.print("[" + puzzleMatrixString[i]
[j] + "]");
            }

            System.out.print("    ");

            for( int j = 0; j < 5; j++){
                if( puzzleMatrixQuestionNumbers[i][j] == null)
{
                    puzzleMatrixQuestionNumbers[i][j] = " ";
                }

                System.out.print("[" +
puzzleMatrixQuestionNumbers[i][j] + "]");
```

```
            }
            System.out.println("");
        }
    }


    private ArrayList<String[]> allSolutionInformation(){

        int currQuestion = -1;
        String currQuestionSol = "";

        ArrayList<String[]> returnThis = new
ArrayList<String[]>();

        String[][] fetchDown = new String[5][5];
        String[][] fetchAccross = new String[5][5];

        for( int i = 0; i < 5; i++){
            for( int j = 0; j<5; j++){
               fetchDown[i][j] =
puzzleMatrixQuestionNumbers[i][j];
               fetchAccross[i][j] =
puzzleMatrixQuestionNumbers[i][j];
            }
        }



        for( int i = 0; i < 5; i++){
            for( int j = 0; j < 5; j++){
                if( !fetchDown[i][j].equals("B") && !
fetchDown[i][j].equals(" ") && !fetchDown[i][j].equals("~")){
```

```
                    if( currQuestion !=
Integer.parseInt(fetchDown[i][j])){
                        currQuestion =
Integer.parseInt(fetchDown[i][j]);
                        currQuestionSol = "";

                        for( int z = i; z<5 && !fetchDown[z]
[j].equals("B") && !fetchDown[z][j].equals("~") ; z++){
                            currQuestionSol = currQuestionSol
+ puzzleMatrixString[z][j];
                            fetchDown[z][j] = "~";
                        }

                        //System.out.println("[DOWN   ]For
question: [" + currQuestion + "] the solution is: [" +
currQuestionSol + "]");
                        String[] temp = new String[3];
                        temp[0] = currQuestion + "";
                        temp[1] = "Down";
                        temp[2] = currQuestionSol;

                        returnThis.add(temp);



                    }
                }
            }
        }

        for( int i = 0; i < 5; i++){
            for( int j = 0; j < 5; j++){
                if( !fetchAccross[i][j].equals("B") && !
fetchAccross[i][j].equals(" ") && !fetchAccross[i]
[j].equals("~")){
```

```
                    if( currQuestion !=
Integer.parseInt(fetchAccross[i][j])){
                        currQuestion =
Integer.parseInt(fetchAccross[i][j]);
                        currQuestionSol = "";

                        for( int z = j; z<5 && !
fetchAccross[i][z].equals("B") && !fetchAccross[i]
[z].equals("~") ; z++){
                            currQuestionSol = currQuestionSol
+ puzzleMatrixString[i][z];
                            fetchAccross[i][z] = "~";
                        }

                        //System.out.println("[ACCROSS]For
question: [" + currQuestion + "] the solution is: [" +
currQuestionSol + "]");
                        String[] temp = new String[3];
                        temp[0] = currQuestion + "";
                        temp[1] = "Accross";
                        temp[2] = currQuestionSol;

                        returnThis.add(temp);
                    }
                }
            }
        }


        for( int i = 0; i < returnThis.size(); i++){
            String[] temp = returnThis.get(i);
            System.out.println("[" + temp[1] + "] For
question: [" + temp[0] + "] the solution is: [" + temp[2] +
"]");
        }
```

```
        return returnThis;
    }

    }
```

GUI related codes:

Class below searches shows the across and down clues:

```
package UI;

import java.util.ArrayList;
import javax.swing.ImageIcon;
import javax.swing.*;
/**
 *
 * @author skerd
 */
public class AcrossPanel extends javax.swing.JPanel {

    /**
     * Creates new form AcrossPanel
     */

    static ArrayList<Object> acrossClueList;
    static ArrayList<Object> downClueList;
    static ArrayList<Object> newAcrossClueList;
    static ArrayList<Object> newDownClueList;

    static Object[][] acrossTableObject;
    static Object[][] downTableObject;
```

```
    static Object[][] oldAcrossTableObject;
    static Object[][] oldDownTableObject;

    public AcrossPanel(ArrayList<Object> acrossClueList,
ArrayList<Object> downClueList, ArrayList<Object>
newAcrossClueList, ArrayList<Object> newDownClueList) {

        this.acrossClueList = acrossClueList;
        this.downClueList = downClueList;
        this.newAcrossClueList = newAcrossClueList;
        this.newDownClueList = newDownClueList;

        Object[][] newAcrosTable = new
Object[acrossClueList.size()][2];
        Object[][] newDownTable = new
Object[downClueList.size()][2];

        Object[][] acrosTable = new
Object[acrossClueList.size()][2];
        Object[][] downTable = new Object[downClueList.size()]
[2];

        for( int i = 0; i < newAcrossClueList.size(); i++){
            newAcrosTable[i][1] =
((ArrayList<Object>)newAcrossClueList.get(i)).get(0);
            newAcrosTable[i][0] = newAcrosTable[i][1] + " " +
((ArrayList<Object>)newAcrossClueList.get(i)).get(1);
        }

        for( int i = 0; i < newDownClueList.size(); i++){
            newDownTable[i][1] =
((ArrayList<Object>)newDownClueList.get(i)).get(0);
            newDownTable[i][0] = newDownTable[i][1] + " " +
((ArrayList<Object>)newDownClueList.get(i)).get(1);
```

```
        }


        for( int i = 0; i < acrossClueList.size(); i++){
            acrosTable[i][1] =
((ArrayList<Object>)acrossClueList.get(i)).get(0);
            acrosTable[i][0] = acrosTable[i][1] + " " +
((ArrayList<Object>)acrossClueList.get(i)).get(1);
        }

        for( int i = 0; i < downClueList.size(); i++){
            downTable[i][1] =
((ArrayList<Object>)downClueList.get(i)).get(0);
            downTable[i][0] = downTable[i][1] + " " +
((ArrayList<Object>)downClueList.get(i)).get(1);
        }

        acrossTableObject = newAcrosTable;
        downTableObject = newDownTable;

        oldAcrossTableObject = acrosTable;
        oldDownTableObject = downTable;

        initComponents();

    }

    static public void selectQuestionInTable(int id, String
direction){

        if( direction.equals("A")){
            String selectedQuestion = "";
            System.out.println("The Across Table has [" +
acrossClueList.size()  + "] rows!" );
```

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

                int temp = (int)((ArrayList<Object>)
(acrossClueList.get(i))).get(0);

                if( temp == id ){
                    selectedQuestion = temp + " " + (String)
((ArrayList<Object>)(acrossClueList.get(i))).get(1);
                    aTable.setRowSelectionInterval(i, i);
                    break;
                }

            }
            //
PuzzlePanel.questionLabel.setText(selectedQuestion);
            dTable.getSelectionModel().clearSelection();
            System.out.println("The selected question is: [" +
selectedQuestion + "]");
        }
        else{
            String selectedQuestion = "";
            System.out.println("The Across Table has [" +
downClueList.size()  + "] rows!" );

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

                int temp = (int)((ArrayList<Object>)
(downClueList.get(i))).get(0);

                if( temp == id ){
                    selectedQuestion = temp + " " + (String)
((ArrayList<Object>)(downClueList.get(i))).get(1);
                    dTable.setRowSelectionInterval(i, i);
                    break;
                }
```

```
            }
            //
PuzzlePanel.questionLabel.setText(selectedQuestion);
            aTable.getSelectionModel().clearSelection();
            System.out.println("The selected question is: [" +
selectedQuestion + "]");
        }

    }
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        acrossPanel = new javax.swing.JPanel();
        jLabel3 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        aTable = new javax.swing.JTable();
        downPanel = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jScrollPane2 = new javax.swing.JScrollPane();
        dTable = new javax.swing.JTable();
        acrossPanel1 = new javax.swing.JPanel();
        jLabel4 = new javax.swing.JLabel();
        jScrollPane3 = new javax.swing.JScrollPane();
        oaTable = new javax.swing.JTable();
        downPanel1 = new javax.swing.JPanel();
        jLabel2 = new javax.swing.JLabel();
        jScrollPane4 = new javax.swing.JScrollPane();
        odTable = new javax.swing.JTable();
```

```
acrossPanel.setBorder(javax.swing.BorderFactory.createLineBord
er(new java.awt.Color(0, 0, 0), 3));

        jLabel3.setFont(new java.awt.Font("Dialog", 1,
14)); // NOI18N
        jLabel3.setText("Across:");

        aTable.setModel(new
javax.swing.table.DefaultTableModel(
            acrossTableObject,
            new String [] {
                ""
            }
        ));
        aTable.setEditingColumn(0);
        aTable.setEditingRow(0);
        aTable.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent
evt) {
                aTableMouseClicked(evt);
            }
        });
        jScrollPane1.setViewportView(aTable);
        aTable.setRowSelectionAllowed(false);

        javax.swing.GroupLayout acrossPanelLayout = new
javax.swing.GroupLayout(acrossPanel);
        acrossPanel.setLayout(acrossPanelLayout);
        acrossPanelLayout.setHorizontalGroup(

acrossPanelLayout.createParallelGroup(javax.swing.GroupLayout.
Alignment.LEADING)
```

```
            .addComponent(jLabel3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addGroup(acrossPanelLayout.createSequentialGroup(
)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 556, Short.MAX_VALUE)
                .addContainerGap())
        );
        acrossPanelLayout.setVerticalGroup(

acrossPanelLayout.createParallelGroup(javax.swing.GroupLayout.
Alignment.LEADING)
            .addGroup(acrossPanelLayout.createSequentialGroup(
)
                .addGap(0, 0, 0)
                .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 18,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAU
LT_SIZE, Short.MAX_VALUE))
        );


        downPanel.setBorder(new
javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 3,
true));


        jLabel1.setFont(new java.awt.Font("Dialog", 1,
14)); // NOI18N
        jLabel1.setText("Down:");
```

```
        dTable.setModel(new
javax.swing.table.DefaultTableModel(
            downTableObject,
            new String [] {
                ""
            }
        ));
        dTable.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent
evt) {
                dTableMouseClicked(evt);
            }
        });
        jScrollPane2.setViewportView(dTable);
        dTable.setRowSelectionAllowed(false);

        javax.swing.GroupLayout downPanelLayout = new
javax.swing.GroupLayout(downPanel);
        downPanel.setLayout(downPanelLayout);
        downPanelLayout.setHorizontalGroup(

downPanelLayout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
            .addGroup(downPanelLayout.createSequentialGroup()
                .addContainerGap()
                .addGroup(downPanelLayout.createParallelGroup(
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE, 550, Short.MAX_VALUE)
                    .addComponent(jScrollPane2,
javax.swing.GroupLayout.DEFAULT_SIZE, 550, Short.MAX_VALUE))
                .addContainerGap())
        );
        downPanelLayout.setVerticalGroup(
```

```
downPanelLayout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
            .addGroup(downPanelLayout.createSequentialGroup()
                .addGap(0, 0, 0)
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 18,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.DEFAULT_SIZE, 107, Short.MAX_VALUE)
                .addContainerGap())
        );


acrossPanel1.setBorder(javax.swing.BorderFactory.createLineBor
der(new java.awt.Color(0, 0, 0), 3));

        jLabel4.setFont(new java.awt.Font("Dialog", 1,
14)); // NOI18N
        jLabel4.setText("Across (Before change):");

        oaTable.setModel(new
javax.swing.table.DefaultTableModel(
            oldAcrossTableObject,
            new String [] {
                ""
            }
        ));
        oaTable.setEditingColumn(0);
        oaTable.setEditingRow(0);
        oaTable.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent
evt) {
```

```
                oaTableMouseClicked(evt);
            }
        });
        jScrollPane3.setViewportView(oaTable);
        aTable.setRowSelectionAllowed(false);

        javax.swing.GroupLayout acrossPanel1Layout = new
javax.swing.GroupLayout(acrossPanel1);
        acrossPanel1.setLayout(acrossPanel1Layout);
        acrossPanel1Layout.setHorizontalGroup(

acrossPanel1Layout.createParallelGroup(javax.swing.GroupLayout
.Alignment.LEADING)
            .addGroup(acrossPanel1Layout.createSequentialGroup
()
                .addGroup(acrossPanel1Layout.createParallelGro
up(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jScrollPane3,
javax.swing.GroupLayout.DEFAULT_SIZE, 556, Short.MAX_VALUE)
                    .addGroup(acrossPanel1Layout.createSequent
ialGroup()
                        .addContainerGap()
                        .addComponent(jLabel4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                .addContainerGap())
        );
        acrossPanel1Layout.setVerticalGroup(

acrossPanel1Layout.createParallelGroup(javax.swing.GroupLayout
.Alignment.LEADING)
            .addGroup(acrossPanel1Layout.createSequentialGroup
()
                .addGap(0, 0, 0)
```

```
                .addComponent(jLabel4,
javax.swing.GroupLayout.PREFERRED_SIZE, 18,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(jScrollPane3,
javax.swing.GroupLayout.PREFERRED_SIZE, 102,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAU
LT_SIZE, Short.MAX_VALUE))
        );

        downPanel1.setBorder(new
javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 3,
true));

        jLabel2.setFont(new java.awt.Font("Dialog", 1,
14)); // NOI18N
        jLabel2.setText("Down (Before Change):");

        odTable.setModel(new
javax.swing.table.DefaultTableModel(
            oldDownTableObject,
            new String [] {
                ""
            }
        ));

        jScrollPane4.setViewportView(odTable);
        dTable.setRowSelectionAllowed(false);

        javax.swing.GroupLayout downPanel1Layout = new
javax.swing.GroupLayout(downPanel1);
        downPanel1.setLayout(downPanel1Layout);
        downPanel1Layout.setHorizontalGroup(
```

```
downPanel1Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
            .addGroup(downPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(downPanel1Layout.createParallelGroup
(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel2,
javax.swing.GroupLayout.DEFAULT_SIZE, 550, Short.MAX_VALUE)
                    .addComponent(jScrollPane4,
javax.swing.GroupLayout.DEFAULT_SIZE, 550, Short.MAX_VALUE))
                .addContainerGap())
        );
        downPanel1Layout.setVerticalGroup(

downPanel1Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
            .addGroup(downPanel1Layout.createSequentialGroup()
                .addGap(0, 0, 0)
                .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 18,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(jScrollPane4,
javax.swing.GroupLayout.DEFAULT_SIZE, 102, Short.MAX_VALUE)
                .addContainerGap())
        );


        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
```

```
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swi
ng.GroupLayout.Alignment.LEADING)
                    .addComponent(acrossPanel,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(downPanel,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(acrossPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(downPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(acrossPanel,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(downPanel,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
```

```
                .addComponent(acrossPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(downPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAU
LT_SIZE, Short.MAX_VALUE))
        );
    }// </editor-fold>//GEN-END:initComponents

    private void aTableMouseClicked(java.awt.event.MouseEvent
evt) {//GEN-FIRST:event_aTableMouseClicked

        int column = 0;
        int row = aTable.getSelectedRow();
        String value = aTable.getModel().getValueAt(row,
column).toString();
        System.out.println("Clicked on acrossTable: " +
value);

        int questionNr = Integer.parseInt( value.substring(0,
value.indexOf(" ")) );

        //PuzzlePanel.questionLabel.setText(value);
        PuzzlePanel.direction = "A";
        dTable.getSelectionModel().clearSelection();

        PuzzlePanel.findQuestion( questionNr );

    }
```

```
    // Variables declaration - do not modify//GEN-
BEGIN:variables
    public static javax.swing.JTable aTable;
    private javax.swing.JPanel acrossPanel;
    private javax.swing.JPanel acrossPanel1;
    public static javax.swing.JTable dTable;
    private javax.swing.JPanel downPanel;
    private javax.swing.JPanel downPanel1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JScrollPane jScrollPane4;
    public static javax.swing.JTable oaTable;
    public static javax.swing.JTable odTable;
    // End of variables declaration//GEN-END:variables
}
```

Class below shows Loading Panel while the puzzle is being downloaded from the website:

```
package UI;

import aiproject.Cs461;
import java.util.ArrayList;
import java.util.concurrent.TimeUnit;
import java.util.logging.Level;
import java.util.logging.Logger;
```

```
/**
 *
 * @author skerd
 */
public class LoadingPanel extends javax.swing.JPanel {

    public static boolean haveAllQuestions;
    public static ArrayList<Object> aaClues;
    public static ArrayList<Object> ddClues;

    public static ArrayList<Object> newDownClues;
    public static ArrayList<Object> newAccrossClues;

    public static ArrayList<Object> tiles;

    public LoadingPanel() {

        initComponents();


        new Thread(new Runnable() {
            public void run() {
                haveAllQuestions = false;
                int count = 1;

                while( !haveAllQuestions ){

                    if( count %3 == 0){
                        loadingLabel.setText("Loading.  ");
                    }
                    else if( count %3 == 1){
                        loadingLabel.setText("Loading.. ");
                    }
```

```
                    else{
                        loadingLabel.setText("Loading...");
                    }

                    count++;

                    try {
                        TimeUnit.MILLISECONDS.sleep((long)
500.0);
                    } catch (InterruptedException ex) {

Logger.getLogger(LoadingPanel.class.getName()).log(Level.SEVER
E, null, ex);
                    }
                }

                System.out.println(tiles.size() + " at loading
panel !!!!!!!!!!!!! ");
                System.out.println("Trying to switch
panels!!!");
                Cs461.main.setCluesAndStartPuzzle( aaClues,
ddClues, newAccrossClues, newDownClues,  tiles );

            }
        }).start();


    }



    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {
```

```
        jLabel1 = new javax.swing.JLabel();
        loadingLabel = new javax.swing.JLabel();
        errorLabel = new javax.swing.JLabel();

        jLabel1.setFont(new java.awt.Font("Dialog", 1,
36)); // NOI18N

jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENT
ER);
        jLabel1.setText("Loading questions from today's
puzzle! ");

        loadingLabel.setFont(new java.awt.Font("Dialog", 1,
48)); // NOI18N

loadingLabel.setHorizontalAlignment(javax.swing.SwingConstants
.CENTER);
        loadingLabel.setText("!");

        errorLabel.setFont(new java.awt.Font("Dialog", 1,
24)); // NOI18N

errorLabel.setHorizontalAlignment(javax.swing.SwingConstants.C
ENTER);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
```

```
                .addGroup(layout.createParallelGroup(javax.swi
ng.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(errorLabel,
javax.swing.GroupLayout.PREFERRED_SIZE, 906,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(0, 154, Short.MAX_VALUE))
                    .addComponent(loadingLabel,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(43, 43, 43)
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 82,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(loadingLabel,
javax.swing.GroupLayout.PREFERRED_SIZE, 77,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(errorLabel,
javax.swing.GroupLayout.PREFERRED_SIZE, 126,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(218, Short.MAX_VALUE))
        );
    }// </editor-fold>//GEN-END:initComponents
```

```
    // Variables declaration - do not modify//GEN-
BEGIN:variables
    public static javax.swing.JLabel errorLabel;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel loadingLabel;
    // End of variables declaration//GEN-END:variables
}
```

Class below keeps the all UI components in a single frame:

```
package UI;

import aiproject.PuzzleGeometry;
import java.awt.BorderLayout;
import java.util.ArrayList;
import java.util.concurrent.TimeUnit;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
/**
 *
 * @author skerd
 */
public class MainFrame extends javax.swing.JFrame {


    static AcrossPanel allClues ;
    static public JLabel question;
    static public LoadingPanel loading;
    static public PuzzlePanel pP;
```

```
    static public int sec;
    static public int min;
    static public int hours;

    static boolean stopTimer = true;

    public void clearTiles(){
        pP.clearTiles();
    }

    public void fill(int row, int col){
        pP.fill(row, col);
    }

    public void setCluesAndStartPuzzle( ArrayList<Object>
across, ArrayList<Object> down, ArrayList<Object> newAcross,
ArrayList<Object> newDown, ArrayList<Object> tiles){

        allClues  = new AcrossPanel( across , down ,
newAcross, newDown);

        System.out.println(tiles.size() + " at main
panel !!!!!!!!!!!!! ");
        pP = new PuzzlePanel( tiles );//this should be
inforamtion from the matrix

        JPanel all = new JPanel();
        all.setLayout( new BorderLayout() );

        all.add( pP, BorderLayout.CENTER);
        all.add( allClues, BorderLayout.EAST);

        //showStartDialog();
```

```
        this.remove(loading);
        this.add(all, BorderLayout.CENTER);
        this.revalidate();
        this.repaint();

        pP.revealPuzzle();



        //runTimer();

    }
     public void showStartDialog(){

        Object[] options = { "OK" };

        int result = JOptionPane.showOptionDialog(null, "Ready
to get started?", "Ready?",
                JOptionPane.YES_OPTION,
JOptionPane.PLAIN_MESSAGE,
                null, options, options[0]);
        if (result == 0){
            System.out.println("Starting puzzle now!!");
            PuzzlePanel.erasePuzzle();
        }
    }



    public MainFrame() {

        initComponents();

        new Thread(new Runnable() {
            public void run() {
```

```
                System.out.println("Loading screen !");
                loading = new LoadingPanel();
                setLayout(new BorderLayout());
                add(loading, BorderLayout.CENTER);
            }
        }).start();

        new Thread(new Runnable() {
            public void run() {
                PuzzleGeometry getQ = new PuzzleGeometry();
            }
        }).start();



        //setCluesAndStartPuzzle(null, null);

    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_C
LOSE);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGap(0, 400, Short.MAX_VALUE)
```

```
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGap(0, 300, Short.MAX_VALUE)
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents


    // Variables declaration - do not modify//GEN-
BEGIN:variables
    // End of variables declaration//GEN-END:variables
}
```

Class below displays the puzzle:

```
package UI;

import static UI.MainFrame.hours;
import static UI.MainFrame.min;
import static UI.MainFrame.sec;
import java.awt.Dimension;
import java.util.ArrayList;
import java.util.Objects;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

/**
 *
 * @author skerd
 */
```

```
public class PuzzlePanel extends javax.swing.JPanel {

    /**
     * Creates new form PuzzlePanel
     */

    public static Tile[][] tileArray;
    static String direction;
    static int lastRow;
    static int lastCol;
    static boolean clicedTwice;

    public PuzzlePanel( ArrayList<Object> tileInfo ) {
        initComponents();

        datee.setText( java.time.LocalDate.now() + "
PowerPuzz" );

        direction = "D";

        tileArray = new Tile[5][5];
        int count = 0;

        System.out.println(tileInfo.size() + " at puzzle panel
!!!!!!!!!!!!! ");

        for( int i = 0; i < 5 ; i++){
            for( int j = 0; j < 5; j++){

                ArrayList<Object> ti = (ArrayList<Object>)
tileInfo.get(count);

                int tn = (int) ti.get(0);
                String tl = (String) ti.get(1);
```

```
                int r = (int) ti.get(2);
                int c = (int) ti.get(3);
                boolean en = (boolean) ti.get(4);
                String cl = (String) ti.get(5);

                //int tileNumber, String tileLetter, int row,
int col, boolean enabled, String correctLetter
                tileArray[i][j] = new Tile( tn, tl, r, c, en,
cl  );
                tileArray[i][j].setPreferredSize(new
Dimension(100,100));
                puzzlePanel.add( tileArray[i][j] );
                count++;
            }
        }
    }


    public static void findQuestion(int questionNr){

        for( int i = 0; i < 5 ; i++){
            for( int j = 0; j < 5; j++){
                if( tileArray[i][j].isTileEnabled() &&
tileArray[i][j].getTileNumber() == questionNr ){
                    System.out.println("Found the tile at: ["
+ i + "," + j + "]");
                    tileArray[i][j].selectTile( i, j );
                    return;
                }
            }
        }

    }
```

```
    public void clearTiles(){
        for( int i = 0; i < 5 ; i++){
            for( int j = 0; j < 5; j++){
                if( tileArray[i][j].isTileEnabled() )
                    tileArray[i]
[j].setBackground(StaticVars.free);
            }
        }
    }

    public static void isPuzzleComplete(){
        boolean isComplete = true;
        for( int i = 0; i < 5 && isComplete; i++){
            for( int j = 0; j < 5; j++){
                if( !tileArray[i][j].isTileCorrect() &&
tileArray[i][j].isTileEnabled()){
                    isComplete = false;
                    break;
                }
            }
        }

        if( isComplete ){

            MainFrame.stopTimer = false;
            JFrame popUpFrame = new JFrame();
            //JOptionPane.showMessageDialog( popUpFrame, "The
puzzle is completed in " + TimeLabel.getText().replace("Time:
", "") );
        }
        else{
            System.out.println("The puzzle is not yet
completed!!");
        }
```

```
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        puzzlePanel = new javax.swing.JPanel();
        datee = new javax.swing.JLabel();

        setMinimumSize(new java.awt.Dimension(607, 641));


puzzlePanel.setBorder(javax.swing.BorderFactory.createLineBord
er(new java.awt.Color(0, 0, 0), 3));
        puzzlePanel.setLayout(new java.awt.GridLayout(5, 5));

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
            .addComponent(puzzlePanel,
javax.swing.GroupLayout.PREFERRED_SIZE, 595,
javax.swing.GroupLayout.PREFERRED_SIZE)
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
```

```
                .addContainerGap()
                .addComponent(puzzlePanel,
javax.swing.GroupLayout.DEFAULT_SIZE, 595, Short.MAX_VALUE))
        );


datee.setHorizontalAlignment(javax.swing.SwingConstants.CENTER
);
        datee.setText("jLabel1");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swi
ng.GroupLayout.Alignment.LEADING, false)
                    .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(datee,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap(javax.swing.GroupLayout.DEFAU
LT_SIZE, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(0, 0, 0)
```

```
                .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(datee,
javax.swing.GroupLayout.PREFERRED_SIZE, 28,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAU
LT_SIZE, Short.MAX_VALUE))
        );
    }// </editor-fold>//GEN-END:initComponents


    // Variables declaration - do not modify//GEN-
BEGIN:variables
    private javax.swing.JLabel datee;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel puzzlePanel;
    // End of variables declaration//GEN-END:variables
}

package UI;
import java.awt.Color;

/**
 *
 * @author skerd
 */
public class StaticVars {

    static Color selected = Color.yellow;
    static Color nearSelected = Color.cyan;
    static Color free = Color.white;
```

```
    static Color disabled = Color.black;
    static Color revealed = Color.orange;
}
```

Class below displays a tile of in the puzzle:

```
package UI;

import java.awt.Color;
import aiproject.Cs461;

/**
 *
 * @author skerd
 */
public class Tile extends javax.swing.JPanel {

    /**
     * Creates new form Tile
     */
    private int tileNumber ;
    private String tileLetter;
    private int row;
    private int col;
    private boolean enabled;
    private boolean revealed;
    private String correctLetter;
    private String tileNumberString;

    public Tile(int tileNumber, String tileLetter, int row,
int col, boolean enabled, String correctLetter) {
        this.tileNumber = tileNumber;
        this.tileLetter = tileLetter;
        this.row = row;
        this.col = col;
```

```
        this.enabled = enabled;
        this.correctLetter = correctLetter;

        if( tileNumber == -1){
            tileNumberString = " ";
        }
        else{
            tileNumberString = this.tileNumber + "";
        }

        initComponents();

        if( !this.enabled ){
            this.setBackground(StaticVars.disabled);
        }



    }

    /**
     * This method is called from within the constructor to
initialize the form.
     * WARNING: Do NOT modify this code. The content of this
method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        RevealedLabel = new javax.swing.JLabel();
        TileNumber = new javax.swing.JLabel();
```

```
        TileLetter = new javax.swing.JLabel();


RevealedLabel.setHorizontalAlignment(javax.swing.SwingConstant
s.RIGHT);
        RevealedLabel.setEnabled(false);


setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(153, 153, 153)));
        setDoubleBuffered(false);
        setMaximumSize(new java.awt.Dimension(75, 75));
        setMinimumSize(new java.awt.Dimension(75, 75));
        setPreferredSize(new java.awt.Dimension(75, 75));
        setRequestFocusEnabled(false);
        addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent
evt) {
                formMouseClicked(evt);
            }
        });
        addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyReleased(java.awt.event.KeyEvent
evt) {
                formKeyReleased(evt);
            }
        });


        TileNumber.setFont(new java.awt.Font("Dialog", 1,
15)); // NOI18N
        TileNumber.setText(tileNumberString + ""
        );


        TileLetter.setFont(new java.awt.Font("Dialog", 1,
80)); // NOI18N
```

```
TileLetter.setHorizontalAlignment(javax.swing.SwingConstants.C
ENTER);
        TileLetter.setText(tileLetter);
        TileLetter.setAlignmentY(0.0F);
        TileLetter.setIconTextGap(0);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addComponent(TileLetter,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addGroup(layout.createSequentialGroup()
                .addComponent(TileNumber,
javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 54, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(TileNumber)
                .addPreferredGap(javax.swing.LayoutStyle.Compo
nentPlacement.RELATED)
                .addComponent(TileLetter,
javax.swing.GroupLayout.PREFERRED_SIZE, 72,
javax.swing.GroupLayout.PREFERRED_SIZE))
        );
    }// </editor-fold>//GEN-END:initComponents
```

```
    private void formMouseClicked(java.awt.event.MouseEvent
evt) {//GEN-FIRST:event_formMouseClicked
        selectTile(this.row, this.col);
    }//GEN-LAST:event_formMouseClicked

    private void formKeyReleased(java.awt.event.KeyEvent evt)
{//GEN-FIRST:event_formKeyReleased

        if( enabled ){
            String letterPressed =  evt.getKeyText((Integer)
evt.getKeyCode() );
            if( letterPressed.length() <= 1){
                System.out.println("Pressed: '" +
letterPressed + "' at tile: [" + this.row + "," + this.col +
"]." );

                if( !isRevealed() ){
                    this.tileLetter =  letterPressed ;
                    this.TileLetter.setText(this.tileLetter);
                }

                selectNextTile(this.row, this.col);
                PuzzlePanel.isPuzzleComplete();
            }
        }

    }//GEN-LAST:event_formKeyReleased

    public void selectTile( int row, int col ){

        Tile temp = PuzzlePanel.tileArray[row][col];

        if( temp.isTileEnabled() ){
```

```
            temp.requestFocus();
            temp.grabFocus();

            Cs461.main.clearTiles();
            Cs461.main.fill(temp.row, temp.col);

            PuzzlePanel.lastRow = temp.row;
            PuzzlePanel.lastCol = temp.col;

            int questionId =
getBelongingQuestion( PuzzlePanel.direction );

            String wordDirection = "";
            if( PuzzlePanel.direction.equals("A") )
                wordDirection = "Across";
            else{
                wordDirection = "Down";
            }

            AcrossPanel.selectQuestionInTable(questionId,
PuzzlePanel.direction);

            System.out.println("Clicked Tile: [" + temp.row +
"," + temp.col + "]. HasFocus: [" + temp.hasFocus() + "] and
belongs to question: [" + questionId + "] with [" +
wordDirection + "] alignment" );

        }
        else{
            System.out.println("The tile is not enabled!");
        }

    }
```

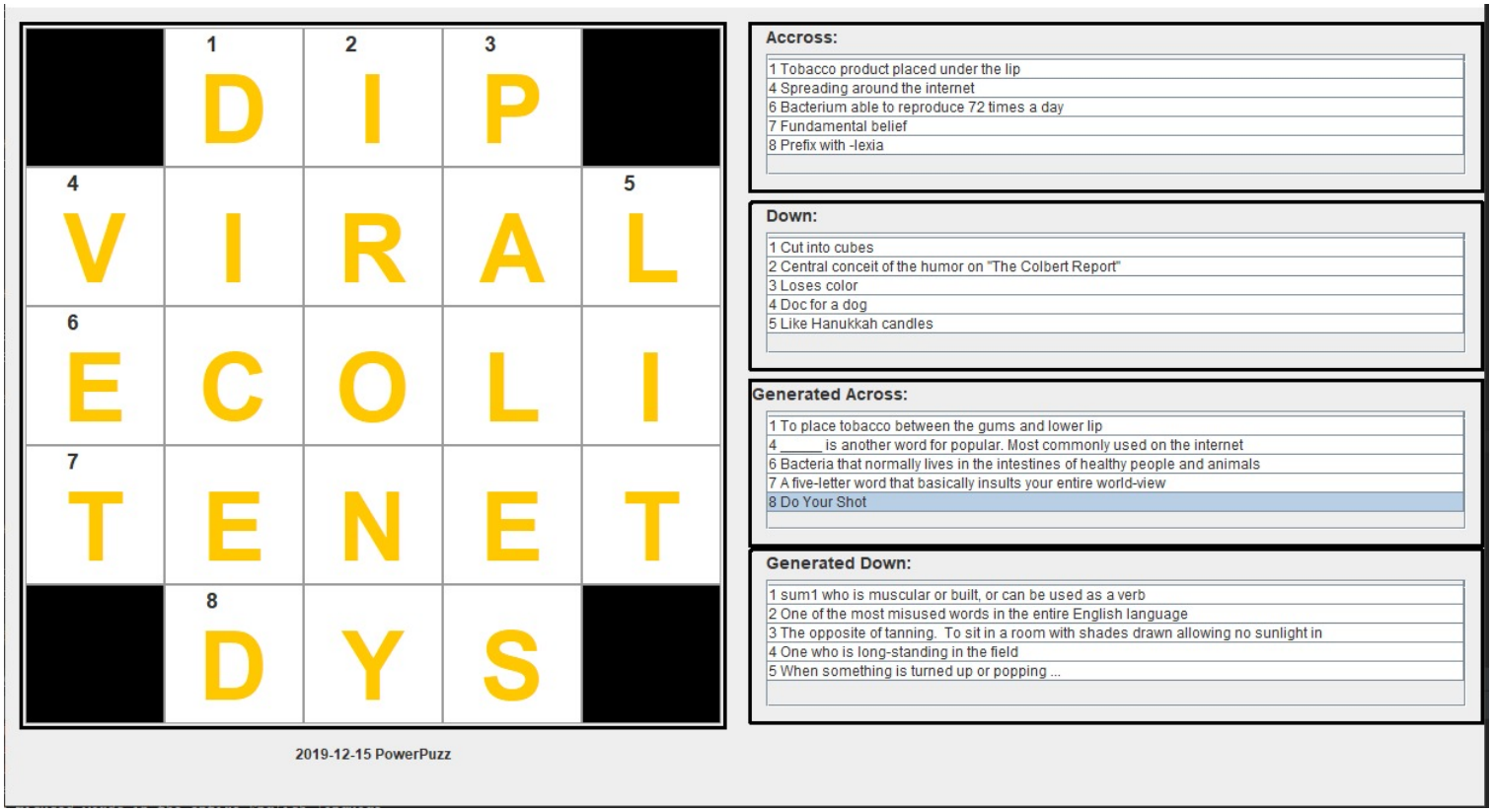
```
    private void selectNextTile(int row, int col){

        if( PuzzlePanel.direction.equals("A")){
            int i = 1;
            int colTemp = (col + 1) % 5  ;
            while(  i <= 5 ){
                if( PuzzlePanel.tileArray[row]
[colTemp].isTileEnabled() ){
                    selectTile(row,colTemp);
                    break;
                }
                colTemp = (colTemp + 1) % 5  ;
                i++;
            }
        }
        else{
            int i = 1;
            int rowTemp = (row + 1) % 5  ;
            while(  i <= 5 ){
                if( PuzzlePanel.tileArray[rowTemp]
[col].isTileEnabled() ){
                    selectTile(rowTemp,col);
                    break;
                }
                rowTemp = (rowTemp + 1) % 5  ;
                i++;
            }
        }
    }

    private int getBelongingQuestion( String direction ){
        if( direction.equals("A")){
            for( int i = 0; i < 5; i++){
```

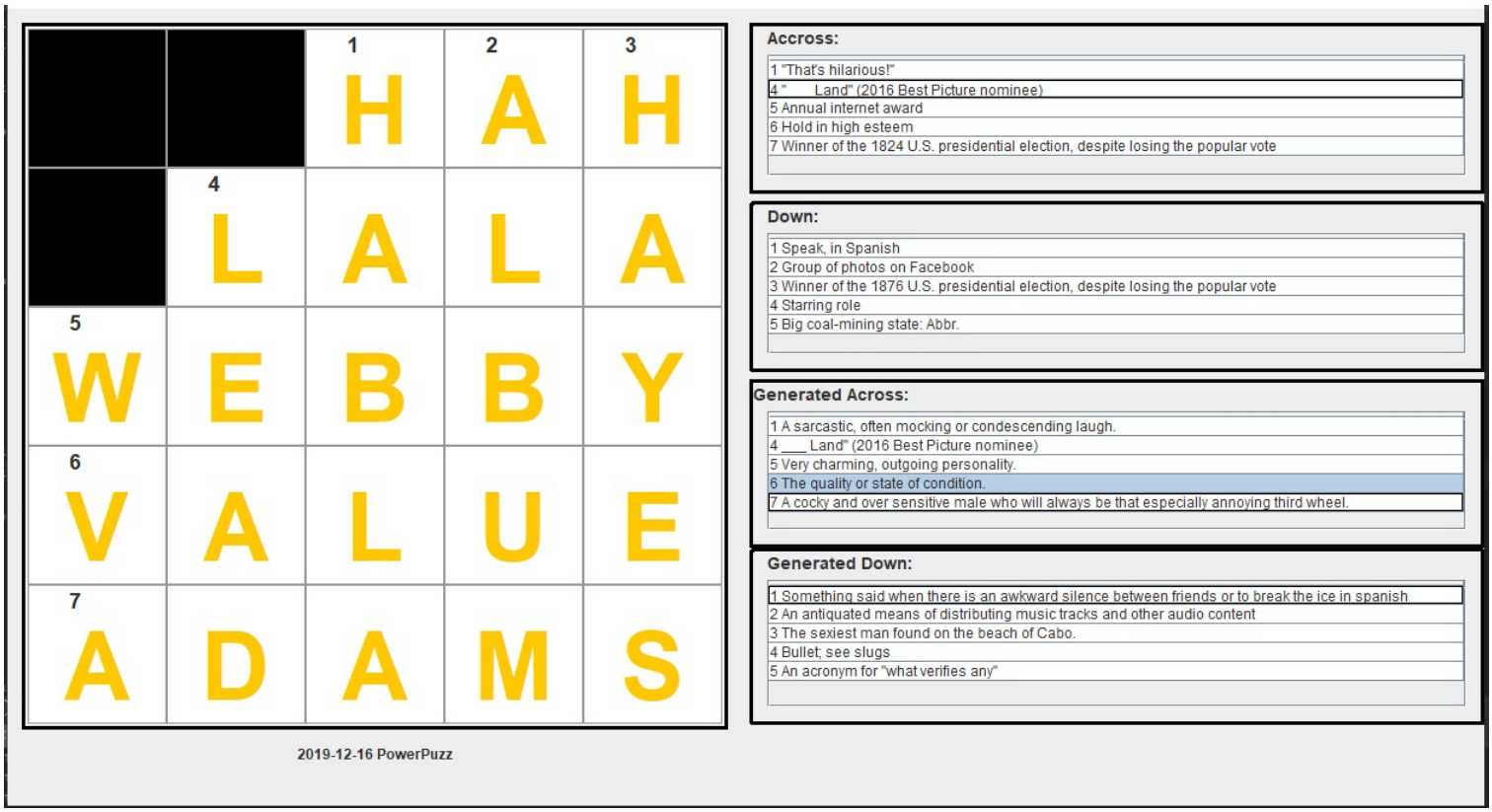
```
                int questionNr =
PuzzlePanel.tileArray[ this.row ][i].getTileNumber();
                if( questionNr != -1  ){
                    return questionNr;
                }
            }
        }
        else if( direction.equals("D")){

            for( int i = 0; i < 5; i++){
                int questionNr = PuzzlePanel.tileArray[i]
[ this.col ].getTileNumber();
                if( questionNr != -1  ){
                    return questionNr;
                }
            }
        }
        return -1;
    }

    public int getTileNumber(){
        return this.tileNumber;
    }

    public int getTileRow(){
        return this.row;
    }

    public int getTileCol(){
        return this.col;
    }

    private String getLetter(){
```

```
    return this.tileLetter;
}

private String getCorrectLetter(){
    return this.correctLetter;
}

public boolean isTileCorrect(){
    if( getLetter().equals( getCorrectLetter() ) ){
                return true;
    }
    else{
        return false;
    }
}

public boolean isTileEnabled(){
    return this.enabled;
}

public boolean isRevealed(){
    return this.revealed;
}

public void clearTile(){
    if( !isRevealed() ){
        this.tileLetter = "";
        this.TileLetter.setText("");
    }
}

public void hardClearTile(){
    this.tileLetter = "";
```

```
        this.TileLetter.setText("");
        this.revealed = false;
        this.RevealedLabel.setText("");
        this.TileLetter.setForeground(Color.black);
    }

    public void revealTile(){

        if( isTileEnabled() ){
            this.tileLetter = this.correctLetter;
            this.TileLetter.setText(correctLetter);

this.TileLetter.setForeground(StaticVars.revealed);
            this.RevealedLabel.setText("R");
            this.revealed = true;
        }
    }



    // Variables declaration - do not modify//GEN-
BEGIN:variables
    private javax.swing.JLabel RevealedLabel;
    private javax.swing.JLabel TileLetter;
    private javax.swing.JLabel TileNumber;
    // End of variables declaration//GEN-END:variables
}
```

Screenshots:





2019-12-17 PowerPuzz

Accross:

1 Spades, hearts, diamonds or clubs
5 "Beats me"
6 Colorado ski resort
7 Frodo and Bilbo's home, with "the"
8 "It rhymes with mint," for this answer

Down:

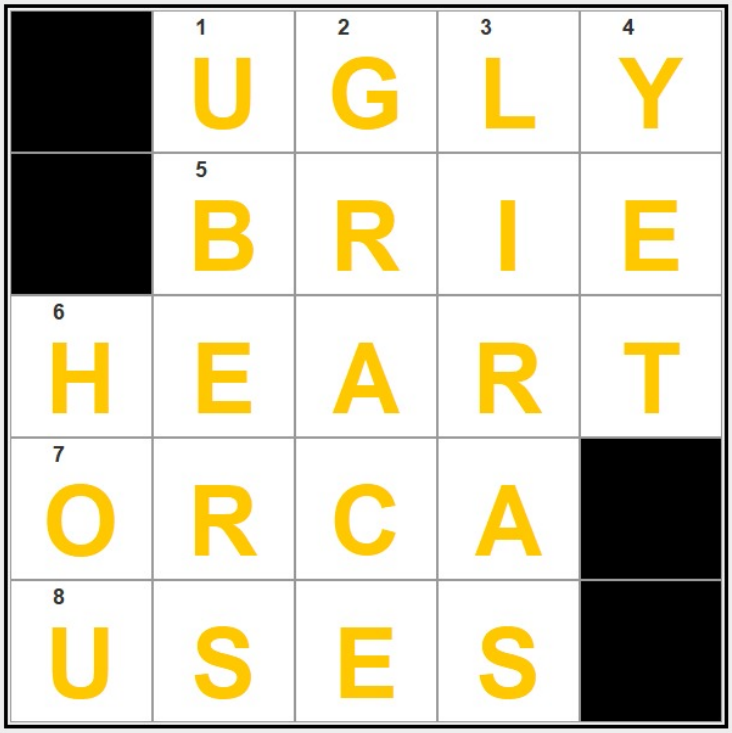
1 Seaweed-wrapped roll
2 Remove, as a corsage
3 Chemically nonreactive
4 Voicemail prompt
5 "Now ___ away! ___ away! ___ away all!" (line in "A Visit From St. Nicholas")

Generated Across:

1 Slang for a businessman or any authority figure
5 Short for Don't Know
6 Amazing cute girl thats fun to talk to
7 A county
8 To uncover information about a certain subject without revealing it in its entirety.

Generated Down:

1 A Japanese dish made with a lump of rice and sweetened vinegar
2 Recode
3 What the balls are.
4 A musical or vocal sound
5 Short for balderdash, which means nonsense. usually a dismissive term.



2019-12-18 PowerPuzz

Accross:

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

Generated Across:

1 Something society trains us to think we are.
5 ____ is a name for a girl who is , sexy, hot,and naturally funny
6 The final element needed to summon Captain Planet!
7 "Killer Whale"
8 Who all your base are belong to

Generated Down:

1 A group of people that are the hardcore users of a system
2 A extremely beautiful and lovable girl who is very talented and intellegent.
3 The definiton of beautiful
4 The word "___" is a word that can change your life!
6 Him over you

2019-12-19 PowerPuzz

Accross:

1 😜
5 "Sure, whatever you say"
6 String for fastening a package
7 Sound from a pig
8 Bathroom cabinetful, for short

Down:

1 Mandel who judges on "America's Got Talent"
2 Three of ___ (poker hand)
3 See 6-Down
4 Sailor's agreement
6 With 3-Down, portrayer of Mister Rogers in 2019's "A Beautiful Day in the Neighborhood"

Generated Across:

1 The quickest way to pretend that you're interested and also a reason for you to close his or her IM box.
5 A lie that people tell so others are happy.
6 A word used when playing soccer or football.
7 The noise that a piggy makes.
8 Medications

Generated Down:

1 The act of stealing another person's partner or dating a friend's sibling.
2 Audiovisual
3 A way to describe something awesome or cool
4 Scottish way of saying yes.
6 Any Canadian Goose that approaches you with caution until offered a pretzel.

2019-12-20 PowerPuzz

Accross:

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:

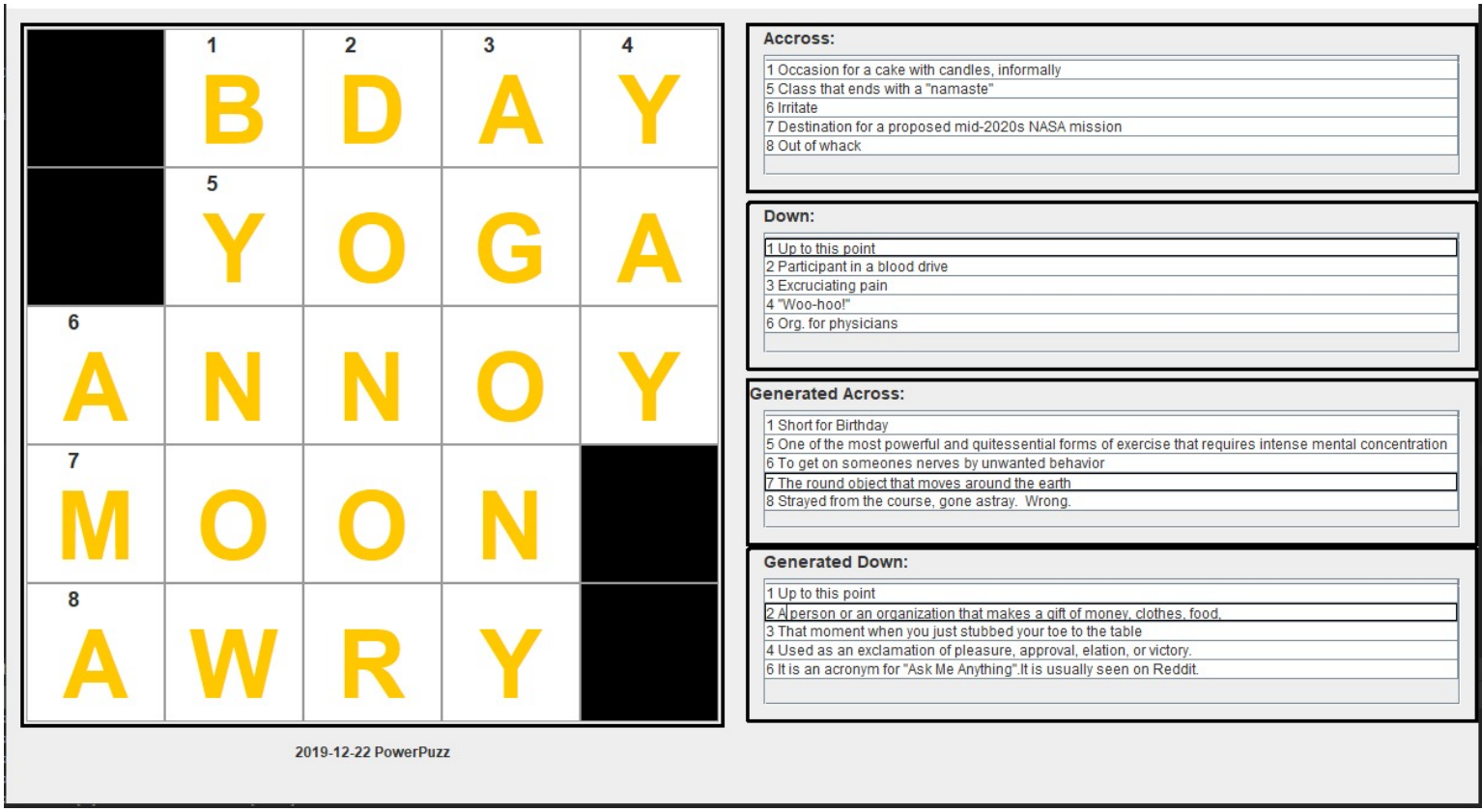
1 Apathetic gesture
2 Group that voted for Trump's impeachment
3 Possessed
4 Colorado has the largest population of this animal in the world, at over 280,000
6 Nonprofit that recruits college grads into education: Abbr.

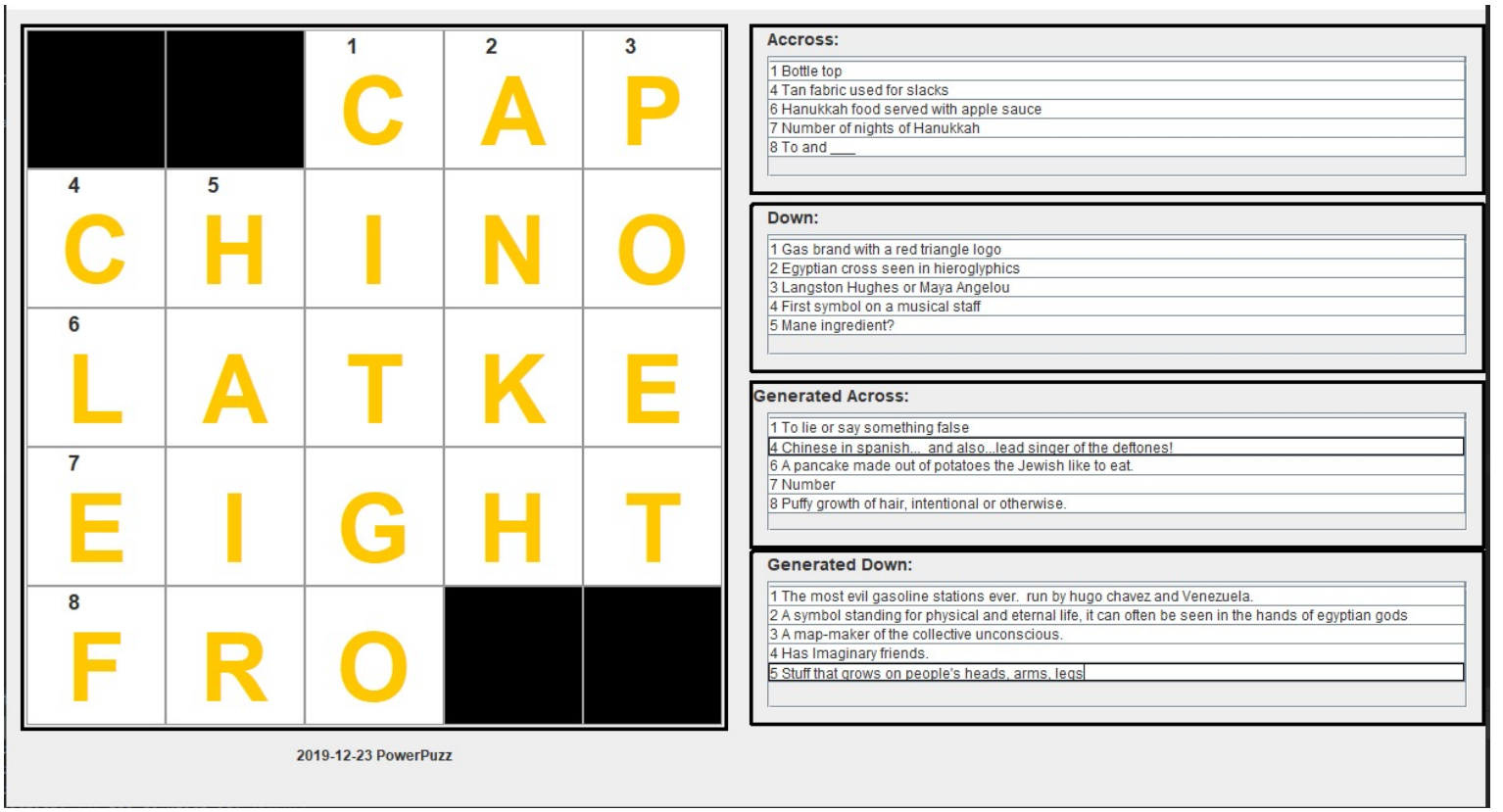
Generated Across:

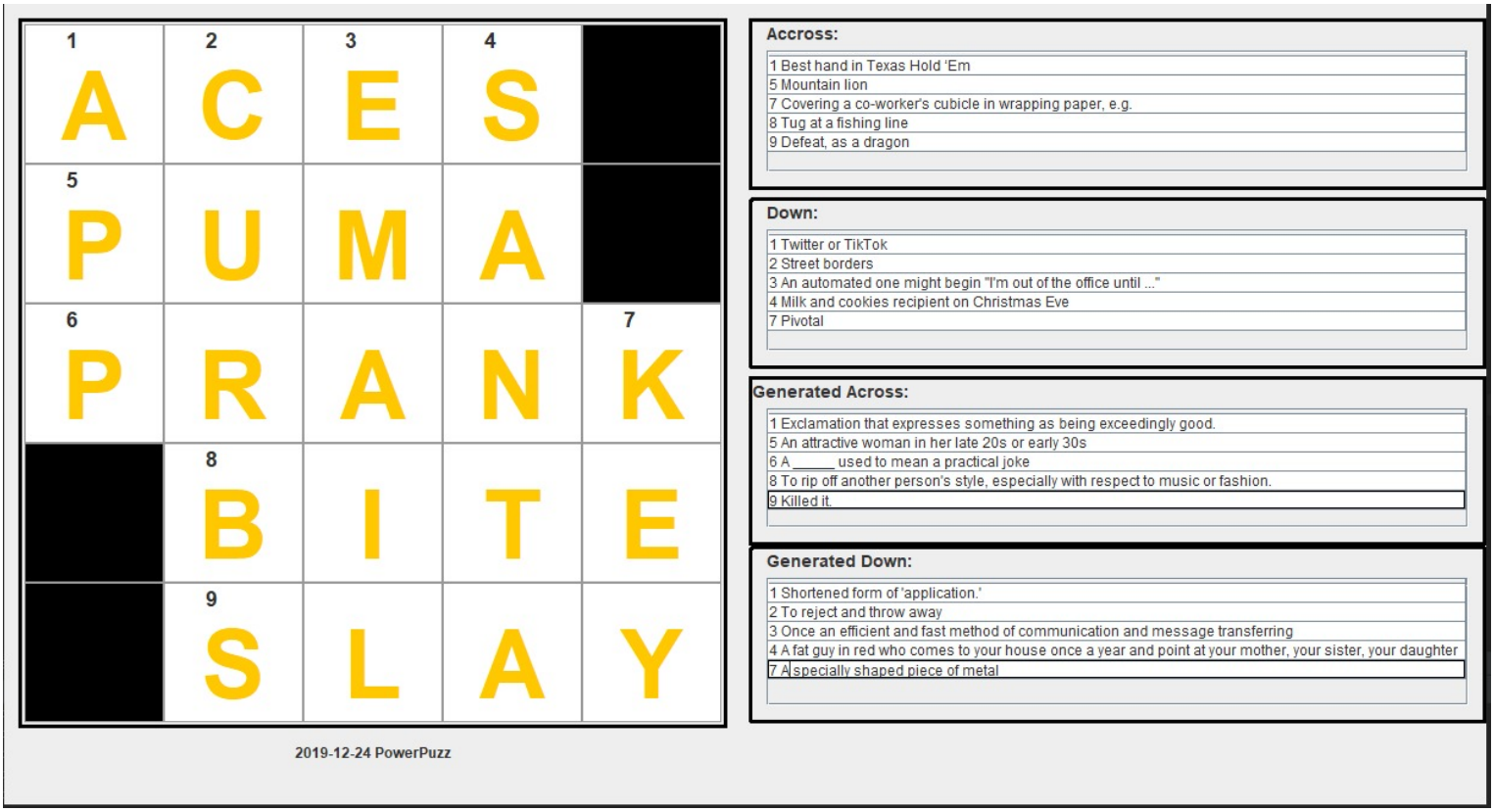
1 A person that is still in the closet, like a ___.
5 It is something humans do at times or some animals.
6 Called the "_____" because its posterior position
7 An electrical device designed to blow breaking the circuit during an overload
8 The slag that the Lord and Savior Gavin has told us

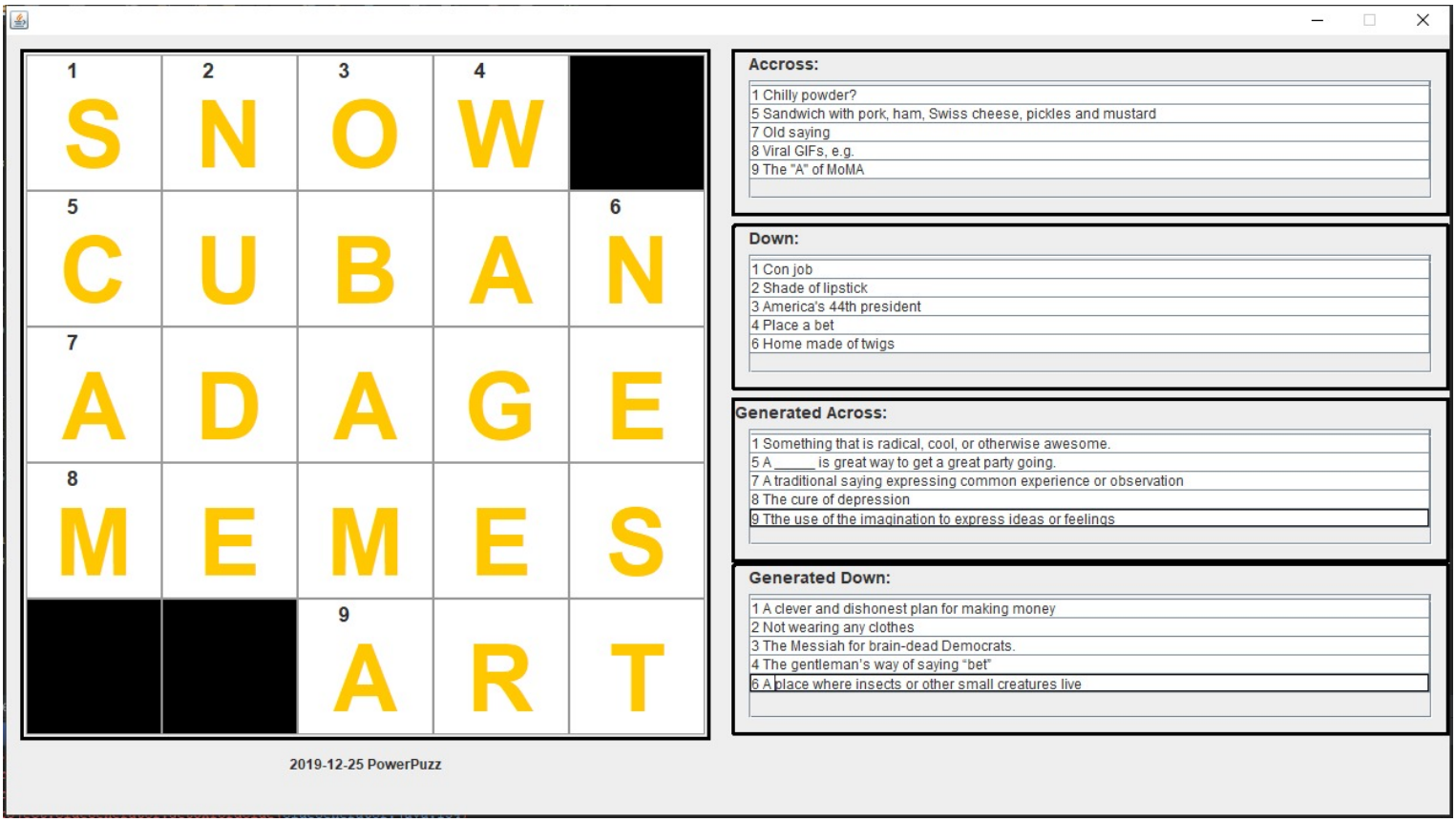
Generated Down:

1 Used to verbally describe someone that is completely irrelevant
2 One of the smartest medical dramas to ever have aired
3 To be made a fool of
4 Large deer like animal standing up to 6 feet tall
6 AFT in the reverse


1 B D A Y
5 Y O G A
6 A N N O Y
7 M O O N
8 A W R Y
Accross:
1 Occasion for a cake with candles, informally
5 Class that ends with a "namaste"
6 Irritate
7 Destination for a proposed mid-2020s NASA mission
8 Out of whack
Down:
1 Up to this point
2 Participant in a blood drive
3 Excruciating pain
4 "Woo-hoo!"
6 Org. for physicians
Generated Across:
1 Short for Birthday
5 One of the most powerful and quitessential forms of exercise that requires intense mental concentration
6 To get on someones nerves by unwanted behavior
7 The round object that moves around the earth
8 Strayed from the course, gone astray. Wrong.
Generated Down:
1 Up to this point
2 A person or an organization that makes a gift of money, clothes, food,
3 That moment when you just stubbed your toe to the table
4 Used as an exclamation of pleasure, approval, elation, or victory.
6 It is an acronym for "Ask Me Anything".It is usually seen on Reddit.
2019-12-22 PowerPuzz


1 C A P
4 C H I N O
5
6 L A T K E
7 E I G H T
8 F R O
Accross:
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?
Generated Across:
1 To lie or say something false
4 Chinese in spanish... and also...lead singer of the deftones!
6 A pancake made out of potatoes the Jewish like to eat.
7 Number
8 Puffy growth of hair, intentional or otherwise.
Generated Down:
1 The most evil gasoline stations ever. run by hugo chavez and Venezuela.
2 A symbol standing for physical and eternal life, it can often be seen in the hands of egyptian gods
3 A map-maker of the collective unconscious.
4 Has Imaginary friends.
5 Stuff that grows on people's heads, arms, legs
2019-12-23 PowerPuzz


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
2019-12-24 PowerPuzz
Accross:
1 Best hand in Texas Hold 'Em
5 Mountain lion
7 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
Generated Across:
1 Exclamation that expresses something as being exceedingly good.
5 An attractive woman in her late 20s or early 30s
6 A _____ used to mean a practical joke
8 To rip off another person's style, especially with respect to music or fashion.
9 Killed it.
Generated Down:
1 Shortened form of 'application.'
2 To reject and throw away
3 Once an efficient and fast method of communication and message transferring
4 A fat guy in red who comes to your house once a year and point at your mother, your sister, your daughter
7 A specially shaped piece of metal


1 S 2 N 3 O 4 W
5 C U B A 6 N
7 A D A G E
8 M E M E S
9 A R T
2019-12-25 PowerPuzz
Accross:
1 Chilly powder?
5 Sandwich with pork, ham, Swiss cheese, pickles and mustard
7 Old saying
8 Viral GIFs, e.g.
9 The "A" of MoMA
Down:
1 Con job
2 Shade of lipstick
3 America's 44th president
4 Place a bet
6 Home made of twigs
Generated Across:
1 Something that is radical, cool, or otherwise awesome.
5 A _____ is great way to get a great party going.
7 A traditional saying expressing common experience or observation
8 The cure of depression
9 Tthe use of the imagination to express ideas or feelings
Generated Down:
1 A clever and dishonest plan for making money
2 Not wearing any clothes
3 The Messiah for brain-dead Democrats.
4 The gentleman's way of saying "bet"
6 A place where insects or other small creatures live