Japanese based Childhood Card Game

By Sein Kim and Serena Wang

Comment since demo: We fixed the issue of the tables having duplicate and instead made a table with keys and now the main tables have numbers that align to another table

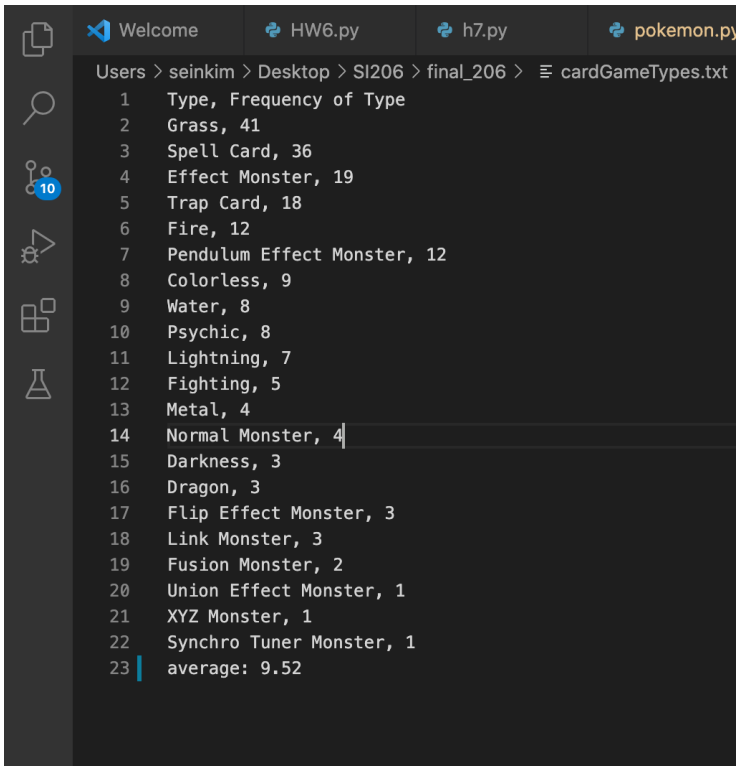
1. The goals for your project (10 points)
    - Among many Japanese games that came to be popular in the United States, Pokemon and Yugioh definitely became huge hits. We wanted to compare these popular games and focus specifically on the deck of cards. The best way we felt to do this was with the different types of cards each of these decks provided. We thought a central theme of these two would be the different types of cards. It was very popular having different types of cards that would essentially achieve different actions. Our primary goal was to identify the different types and see which types were most prevalent in each of the cards. We wanted to compare between Yugioh and Pokemon, the frequency of types and races(for Yugioh) and the average of each type(for Pokemon).
2. The goals that were achieved (10 points)
    - Among the Pokemon cards in a set of 100 cards it seemed that 41 of the cards were grass type. Which was very interesting to see especially since if you search what is the most common type of pokemon, the answer is water. Which was only 8 out of the 100 cards. In terms of rarity it was interesting that in these 100 cards the most common was Rare Holo so this may have to do with the API itself and which cards they included in the API.
    - We also found that out of 100 Yugioh cards, the most common types of cards were Spell cards and Trap cards - monster cards were less common. And the most common race was Normal, which makes sense.
3. The problems that you faced (10 points)
    - Some of the problems we faced included the fact that Pokemon and Yugioh work very differently. While Pokemon has types based on elements, like fire, water, grass, etc., Yugioh has different types of cards (Like spell cards or monster cards) as well as different races (Beast, Normal, etc). So it was impossible to directly compare Pokemon and Yugioh. Additionally, many Yugioh cards but not all have an archetype, which I(Serena) didn't find out until playing with the API a bit more. Before finding out that not all Yugioh cards have an archetype, I was confused as to why my loop wasn't correctly iterating through all the cards.
    - Another problem that was actually using the API got a bit confusing so I had to look up more information on the APIs to actually begin the project.
    - I think the examples were really helpful since the project alone was kind of confusing but after watching the lecture on the examples the instructions became more clear
    - More technical difficulties like matplot and clarifications on the project were handled through piazza and google as listed in the resources below.
4. Your file that contains the calculations from the data in the database (10 points)
    - Yugioh.txt

```
Race, Frequency
Normal, 29
Fiend, 16
Continuous, 10
Equip, 7
Quick-Play, 5
Machine, 5
Fish, 4
Warrior, 4
Aqua, 3
Insect, 3
Beast, 2
Field, 2
Cyberse, 2
Spellcaster, 1
Ritual, 1
Beast-Warrior, 1
Rock, 1
Fairy, 1
Dragon, 1
Sea Serpent, 1
Plant, 1

Type, Frequency
Spell Card, 36
Effect Monster, 19
Trap Card, 18
Pendulum Effect Monster, 12
Normal Monster, 4
Flip Effect Monster, 3
Link Monster, 3
Fusion Monster, 2
Union Effect Monster, 1
XYZ Monster, 1
Synchro Tuner Monster, 1
```

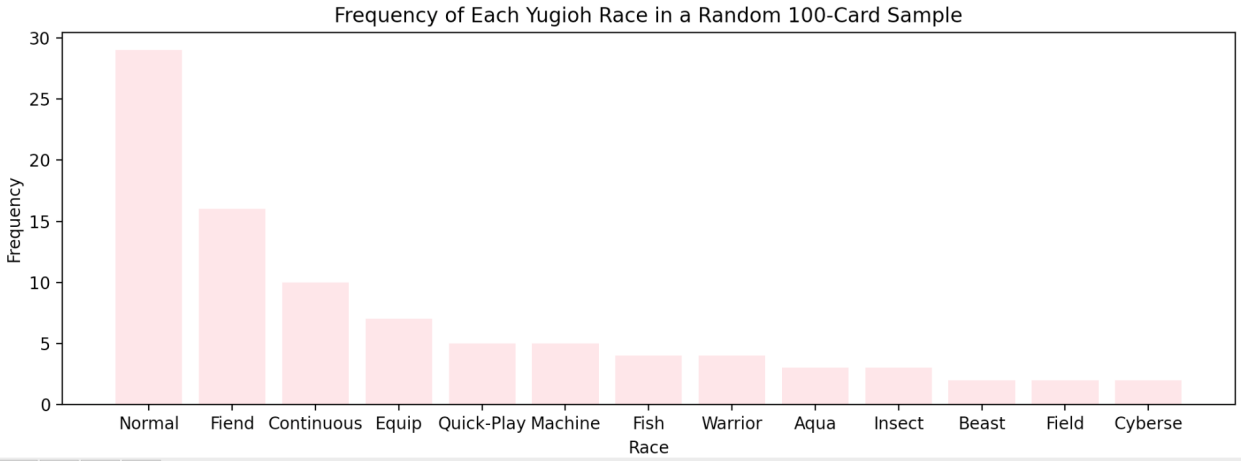
- Pokemon.txt

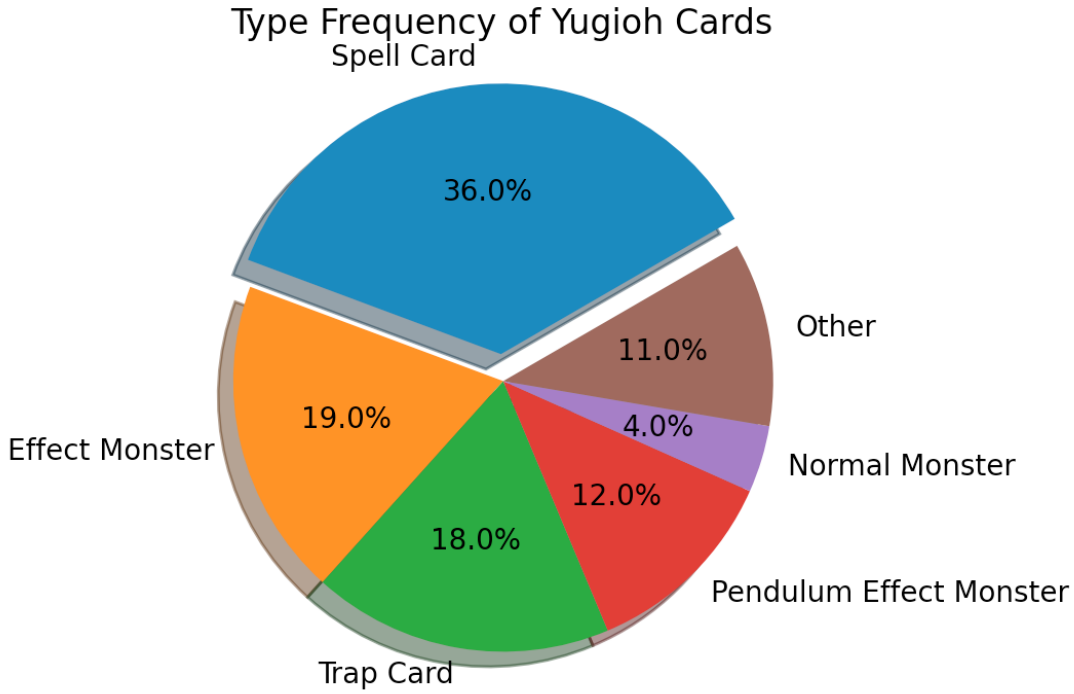
```
Type, Frequency of Type
Grass, 41
Fire, 12
Colorless, 9
Water, 8
Psychic, 8
Lightning, 7
Fighting, 5
Metal, 4
Darkness, 3
Dragon, 3

Rarity, Frequency of Rarity
Rare Holo, 54
Common, 19
Rare, 10
Uncommon, 6
norarity, 5
Promo, 3
Rare Holo GX, 2
Rare Holo V, 1
```

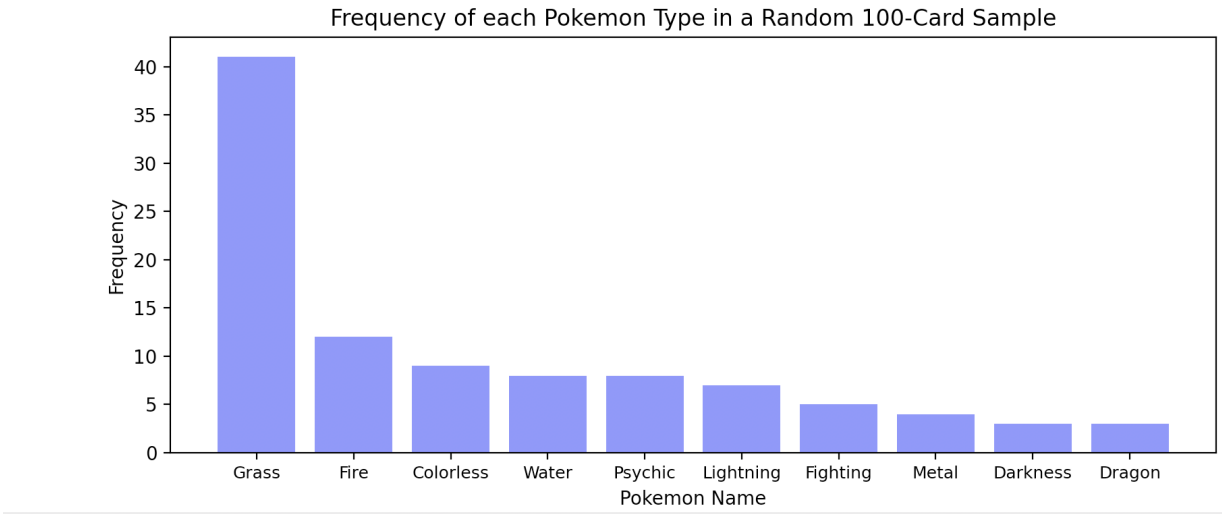
- cardGamesTypes.txt:

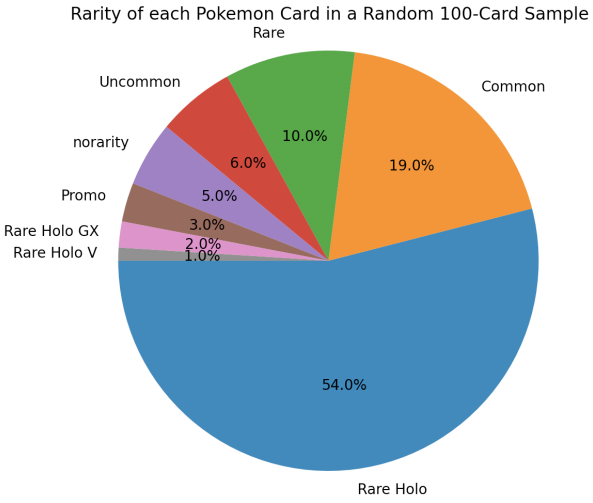


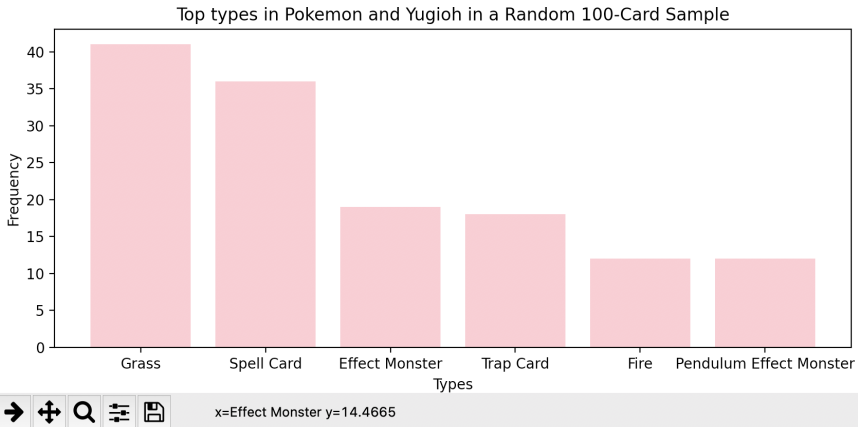
5. The visualization that you created (i.e. screen shot or image file) (10 points)


Frequency of Each Yugioh Race in a Random 100-Card Sample
Frequency
Race
0
5
10
15
20
25
30
Normal
Fiend
Continuous
Equip
Quick-Play
Machine
Fish
Warrior
Aqua
Insect
Beast
Field
Cyberse


Type Frequency of Yugioh Cards
Spell Card
36.0%
Other
11.0%
Effect Monster
19.0%
4.0%
Normal Monster
12.0%
18.0%
Pendulum Effect Monster
Trap Card







6. Instructions for running your code (10 points)
    - Yugioh:

- Set up the database CardGames.db
- Save the result of the cards() function to a variable (data)
- Use setDB(data, cur, conn) and setRaceDB(cur, conn) to set up the 2 tables for Yugioh
- Run getTypeFreq(cur) and getRaceFreq(cur) and save both results to 2 variables (typeDict, raceDict)
- Run barChart(raceDict) and pieChart(typeDict) to get the visualizations
- Run write_to_csv(filename, raceDict, typeDict) to get the written data in another file (I named it Yugioh.txt)

- Pokemon:
  - Delete all tables for Pokemon in CardGames.db
  - Set up the database CardGames.db
  - Uncomment

```
setTypeDB(cur, conn)
    setRarityDB(cur, conn)
    pokemon_list = setUpEpisodes(json_data)
    makeTable(pokemon_list, cur, conn)
```

  - Run code Pokemon.py four times

```
(base) seinkim@Seins-MacBook-Pro-2 final_206 % python3 pokemon.py

----------------------------------------------------------------------
Ran 0 tests in 0.000s

OK
(base) seinkim@Seins-MacBook-Pro-2 final_206 % python3 pokemon.py

----------------------------------------------------------------------
Ran 0 tests in 0.000s

OK
(base) seinkim@Seins-MacBook-Pro-2 final_206 % python3 pokemon.py

----------------------------------------------------------------------
Ran 0 tests in 0.000s

OK
(base) seinkim@Seins-MacBook-Pro-2 final_206 % python3 pokemon.py

----------------------------------------------------------------------
Ran 0 tests in 0.000s

OK
(base) seinkim@Seins-MacBook-Pro-2 final_206 %
```

  - Uncomment these lines of code

```
# barChart(cur)
    # pieChart(cur)
    # writeCSV("pokemon.txt", cur)
    # cardGameDict = writeCSVforboth("cardGameTypes.txt", cur)
    # barChartforBoth(cardGameDict)
```

    - And comment this will ensure that there are only 100 spots

```
setTypeDB(cur, conn)
    setRarityDB(cur, conn)
    pokemon_list = setUpEpisodes(json_data)
    makeTable(pokemon_list, cur, conn)
```

        - Run Pokemon.py and this will make all the charts and the txt files

7. Documentation for each function that you wrote. This includes the input and output for each function (20 points)

- Pokemon.py

```
def createJSON():
    "This function takes the API and turns it in to a JSON which stores the dict that the API provided"

def readDataFromFile(filename):
    "This function takes the JSON created in the first function and reads the file and returns the data in the JSON file"

def setUpDatabase(db_name):
    "This function takes the cardGames.db as a paramater and sets it up to return the cur and conn"

def setUpEpisodes(data):
    '''This function takes the json data we created before and makes a list of all the pokemon cards
    with tuples that include the name,rarity and types of each card. On top of this, this sets an id
    to each pokemon which is determined just by the count of the pokemon,starting from one it gives
    each pokemon a number id. It then takes this information as a list of tuples and returns this list'''

def makeTable(data, cur, conn):
    '''this function makes the table that will go into the data base. it take the data from the JSON
    file and cur and conn created by the setUpDatabase function. This creates the Pokemon cards table
    which has id, name, type and rarity and the Pokemon Type table that just has the types and frequency of type
    and the Pokemon Rarity Table that has the rarity and frequency of rarity. It starts inserting in the
    table by only inserting 25 each time the function is run for the Pokemon Cards it is just inserted
    where as for the pokemon rarity and type it is incremented with the UPDATE function. Each time
    this function is run it takes the last inserted id for the cards table and uses that keep inserting
    new cards each time the function is run'''
def barChart(cur):
    "Using matplot this creates the bar chart for the types and frequency of type in the Pokemon types table"

def pieChart(cur):
    "Using matplot this creates the pie chart for the rarity and frequency of rarity in the Pokemon rarity table"

def writeCSV(filename, cur):
    '''This function takes the filename of pokemon.txt and goes through type and rarity table and creates a txt file
    that lists out all types and rarity and the frequency of each of these values'''

def writeCSVforboth(filename, cur):
    '''This function takes the filename of cardGameTypes.txt and goes through Pokemon_Type and Races table and creates a txt file
    that lists out all types and races in Pokemon and Yugioh and the frequency of each of these values, it also calcultes the average
    of all these values. This function returns a dict that includes all these values'''

def barChartforBoth(cardGameDict):
    '''This function takes the dict from the writeCSVforboth that includes all types and races in Pokemon and
    Yugioh and the frequency of each of these values and creates a bar chart that only hold values that are
    greater than or equal to 10 so we can only get the top values'''

```

-
    - Added def setTypeDB(cur, conn): and def setRarityDB(cur, conn): after the demo we realized that we needed to have a key instead of having duplicate names so these two functions take in cur and conn and create tables for the type and rarity keys that is used in the main card tables
- Yugioh.py
    - def cards(): # Sets up and returns a list of tuples for card info to later insert into the database. Takes nothing as input.
    - def setUpDatabase(db_name): #A function to set up the database where all the information will be stored in. Takes the db_name as input and returns cur, conn.
    - def setDB(yugioh, cur, conn): #Sets overall database, takes the card data from cards() function and returns nothing.
    - def setRaceDB(cur, conn): #Set up a separate database for the races and race ids. Takes cur, conn as input and returns nothing
    - def getRaceFreq(cur): #Use a JOIN to get frequency of races, takes cur as input and returns a dictionary with the race as key and frequency as value

    - def barChart(raceDict): #Creates a barchart based off of the returned race dictionary, takes the output from getRaceFreq as input and returns nothing
    - def getTypeFreq(cur):#Gets frequency of types and takes cur as input, returns a dictionary with keys as types and values as frequencies
    - def write_to_csv(filename, raceDict, typeDict):#Writes all calculated data to a csv file. Takes filename, output values from getRaceFreq and getTypeFreq as input and returns nothing.
- For both main() it just outputs the functions above
    - def pieChart(typeDict):#Creates a pie chart showing the frequency of each type, takes output from getTypeFreq as input and returns nothing.

8. You must also clearly document all resources you used. The documentation should be of the following form (20 points)

Date	Issue Description	Location of Resource	Result
12/6/2021	Wanted a good example to begin starting to better understand the project	https://github.com/elainatiller/public_final_joe_rogan	Yes looking at these examples really helped me better understand the idea of the project and allowed me to start.
12/6/2021	Was not positive how to use the API for Pokemon and where the API key was used	https://www.reddit.com/r/pkmntcg/comments/lfofci/version_2_of_the_pok%C3%A9mon_tcg_api_released/	Learned that I did not need the key I signed up for and with the version two I could just use the API right away and access all the cards
12/7/2021	I was not sure how to get the API into a database so I realized I needed to make it to a JSON first but I was still confused	https://stackoverflow.com/questions/57705844/how-to-convert-an-api-response-to-a-json-object	I realized the correct way to do this was with the dump function and using this post I was able to go back to our past homeworks and fix this issue
12/8/2021	I was not sure to how to increment my count for my table	https://stackoverflow.com/questions/9293900/how-to-increment-integer-columns-value-by-1-in-sql	This post helped me see the UPDATE keyword which then helped me be able to increment all my values
12/9/2021	I had general clarifications on some of the project instructions	https://piazza.com/class/ksgl012i3yw3lz?cid=356	I went directly to piazza and asked multiple questions relating to clarifications of JOIN and single database and my questions were quickly answered by instructors and students
12/9/2021	Was having trouble figuring out how to make a bar chart	https://pythonspot.com/matplotlib-bar-chart/	Yes, I learned all the inputs needed for the barchart function and how to arrange it into a proper bar chart with a

			title, labels and colored bars.
12/9/2021	Wanted to figure out how to make pie chart different from lecture (using explode and starting angle)	https://matplotlib.org/stable/gallery/pie_and_polar_charts/pie_features.html	Yes, I learned all the inputs needed for a piechart and how to arrange it, and also figured out extra stuff such as “explode” and the starting angle which I was able to experiment with.