**Lab 9: The Final Battle Part 2**

CSC115/170 Spring 2025

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

**Objective:**

In this lab you are going to use everything we have talked about this semester to write the code to defeat the evil sorcerer and restore the mystical code - The Syntak - to the capital, save the King and the entire Kingdom. Whew!!! That is a lot.

**Tasks for Lab 9 Part 2**

* Write an opening narrative and save it as **opening##.txt** (where ## is your group number)
* Write a closing narrative and save it as **closing##.txt**
* Use Generative AI to create a function called read\_text\_file with the specifications described in the assignment.
  + Document this process below
* Use a different Generative AI system to create a function called read\_csv\_file with the specifications described
  + Document this process below
* Upload the completed **opening** file, **closing** file and **Word document** to D2L by the due date.

## Partner names:

| Partner 1: Benjamin Lewis  Partner 2: Esten Odney |
| --- |
|  |

## 1. read\_text\_file function

Use Generative AI to write a function that will take a filename as a parameter and return a string of the file contents.  Check if the file exists, if it doesn't, return a blank string.

### Which partner worked on this function?

| **Ben Lewis** |
| --- |

### Which AI system did you choose? Describe why you chose this one. Include a link.

| **https://chatgpt.com**  **I used chatgpt because I have used it before and was familiar with how it worked.** |
| --- |

### What exact prompt did you begin with?

| **Write a python function called read\_text\_file that reads a text file. Here is a list of things the function should do: Take the name of the text file as a parameter, read the text file into a string, make sure the file exists (using try and except ), If the file exists, return the contents of the file as a string, If the file does not exist, return an empty string, and contain a Docstring with name/parameters/returns/and example usage.** |
| --- |

### What was your final prompt that produced the best code (exact prompt)?

| **Write a python function called read\_text\_file that reads a text file. Here is a list of things the function should do: Take the name of the text file as a parameter, read the text file into a string, make sure the file exists (using try and except ), If the file exists, return the contents of the file as a string, If the file does not exist, return an empty string, and contain a Docstring with name/parameters/returns/and example usage.** |
| --- |

### What was the final function produced? Test it with the provided test code below. Make sure your code handles when files are not found.

| *# insert function here*  **def** read\_text\_file(filename):  """  Reads the contents of a text file and returns it as a string.  Parameters:  filename (str): The name or path of the text file to be read.  Returns:  str: The contents of the file as a string if it exists,  otherwise an empty string if the file cannot be found.  Example usage:  contents = read\_text\_file("example.txt")  print(contents)  """  try:  with open(filename, 'r', encoding='utf-8') as file:  return file.read()  except FileNotFoundError:  return "" |
| --- |

**Test code (just copy this in VSCode to test your function):**

filename **=** 'opening##.txt'    *# change this to match your filename*

story **=** read\_text\_file(filename)

**if** story **!=** "":

    print(story)

**else**:

    print("Story is blank")

## 2. read\_csv\_file function

Use Generative AI to write a function that will take a filename of a csv file as a parameter.  This file has a header and two columns.  The first column contains spellnames as strings and the last column contains spell power as numbers.  Return two NumPy arrays - one of the spell names column and another of the spell power column. If the file does not exist, return two empty NumPy arrays.

### Which partner worked on this function?

| Esten |
| --- |

### Which AI system did you choose? Describe why you chose this one. Include a link.

| Google Gemini. This one was simple to set up and use as I already had a gmail account.  https://gemini.google.com/app |
| --- |

### What exact prompt did you begin with?

| Use Generative AI to write a function that will take a filename of a csv file as a parameter.  This file has a header and two columns.  The first column contains spellnames as strings and the last column contains spell power as numbers.  Return two NumPy arrays - one of the spell names column and another of the spell power column. If the file does not exist, return two empty NumPy arrays. |
| --- |

### What was your final prompt that produced the best code (exact prompt)?

| Write a python function called "read\_csv\_file" that will read a csv file into a NumPy array. The function should take a csv file as a parameter. The csv file should have 2 columns - a spell name and damage inflicted. This file will have a header. It will check if the file exists (using try and except ). If the file exists, return the data as two NumPy arrays - one array for the spell name (str) column and one array for the damage (int) column. If the file does not exist, return two empty NumPy arrays. Use a Docstring with name/parameters/returns/and example usage.  After it wrote the code, it created its own sample csv file. To fix this I prompted: "Don't include the test file", and "can you delete the code for testing? I plan on testing the function myself". This eliminated the extra code it generated. |
| --- |

### What was the final function produced? Test it with the provided test code below. If you don’t get the same output, then you need to change your code.

| def read\_csv\_file(filename):  """Reads a CSV file containing spell names and damage into NumPy arrays.  Parameters:  filename (str): The path to the CSV file. The file should have a header row,  with the first column containing spell names (strings) and the  second column containing damage inflicted (integers).  Returns:  tuple: A tuple containing two NumPy arrays:  - spell\_names (np.ndarray): An array of spell names (strings).  - damage (np.ndarray): An array of damage values (integers).  If the file does not exist, returns two empty NumPy arrays.  Example Usage:  # Assume you have a CSV file named 'spells\_data.csv' with the following content:  # spell\_name,damage  # Magic Missile,10  # Fireball,25  # Ice Bolt,15  names, damage\_values = read\_csv\_file('spells\_data.csv')  print("Spell Names:", names)  print("Damage Values:", damage\_values)  names\_empty, damage\_empty = read\_csv\_file('non\_existent.csv')  print("Spell Names (non-existent):", names\_empty)  print("Damage Values (non-existent):", damage\_empty)  """  try:  df = pd.read\_csv(filename)  spell\_names = df.iloc[:, 0].to\_numpy(dtype=str)  damage = df.iloc[:, 1].to\_numpy(dtype=int)  return spell\_names, damage  except FileNotFoundError:  print(f"Error: File '{filename}' not found.")  return np.array([]), np.array([])  except Exception as e:  print(f"An error occurred while reading '{filename}': {e}")  return np.array([]), np.array([]) |
| --- |

**Test code (just copy this in VSCode to test your function):**

*# test code*

*# - file not found*

filename **=** 'nofile.csv'

spellnames, spellpower  **=** read\_csv\_file(filename)

print(**f**"datatypes spellnames:{type(spellnames)}, spellpower:{type(spellpower)}")

**if** np.size(spellnames) **==** 0:

    print("Array is empty\n")

*# - file exists*

filename **=** 'spells.csv'

print(**f**"File: {filename}")

spellnames, spellpower  **=** read\_csv\_file(filename)

print(**f**"datatypes spellnames:{type(spellnames[0])}, spellpower:{type(spellpower[0])}\n")

**for** i **in** range(np.size(spellnames)):

    print(spellnames[i], spellpower[i])

**Output should be:**

File not found: nofile.csv not found.

datatypes spellnames:<class 'numpy.ndarray'>, spellpower:<class 'numpy.ndarray'>

Array is empty

File: spells.csv

datatypes spellnames:<class 'numpy.str\_'>, spellpower:<class 'numpy.int32'>

fireball 50

lightning 25

hail stones 20

shadow blade 20

blight 30

vampiric touch 25

acid arrow 20

***Upload this document and the opening and closing files to D2L before the due date.***