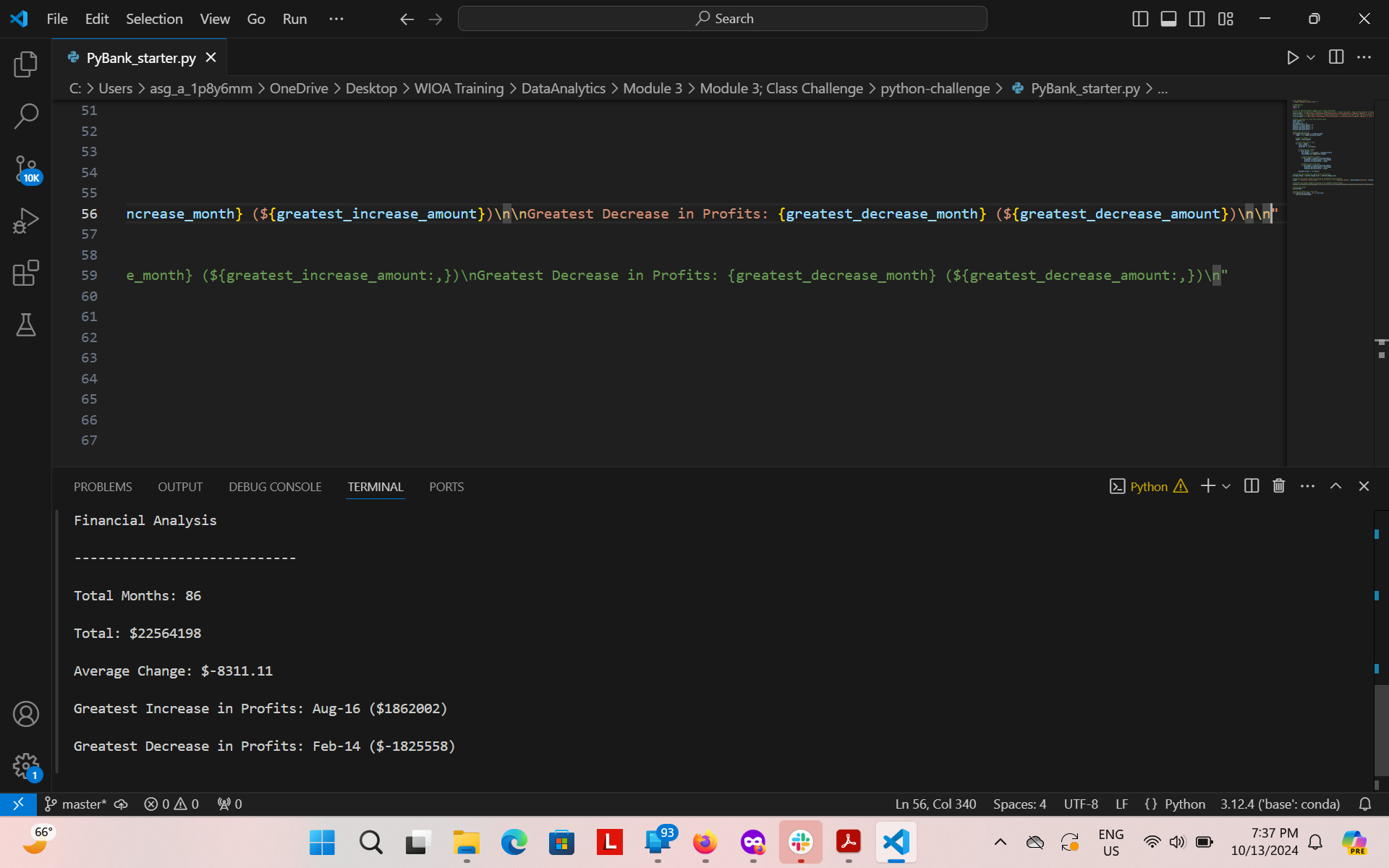
**Main**

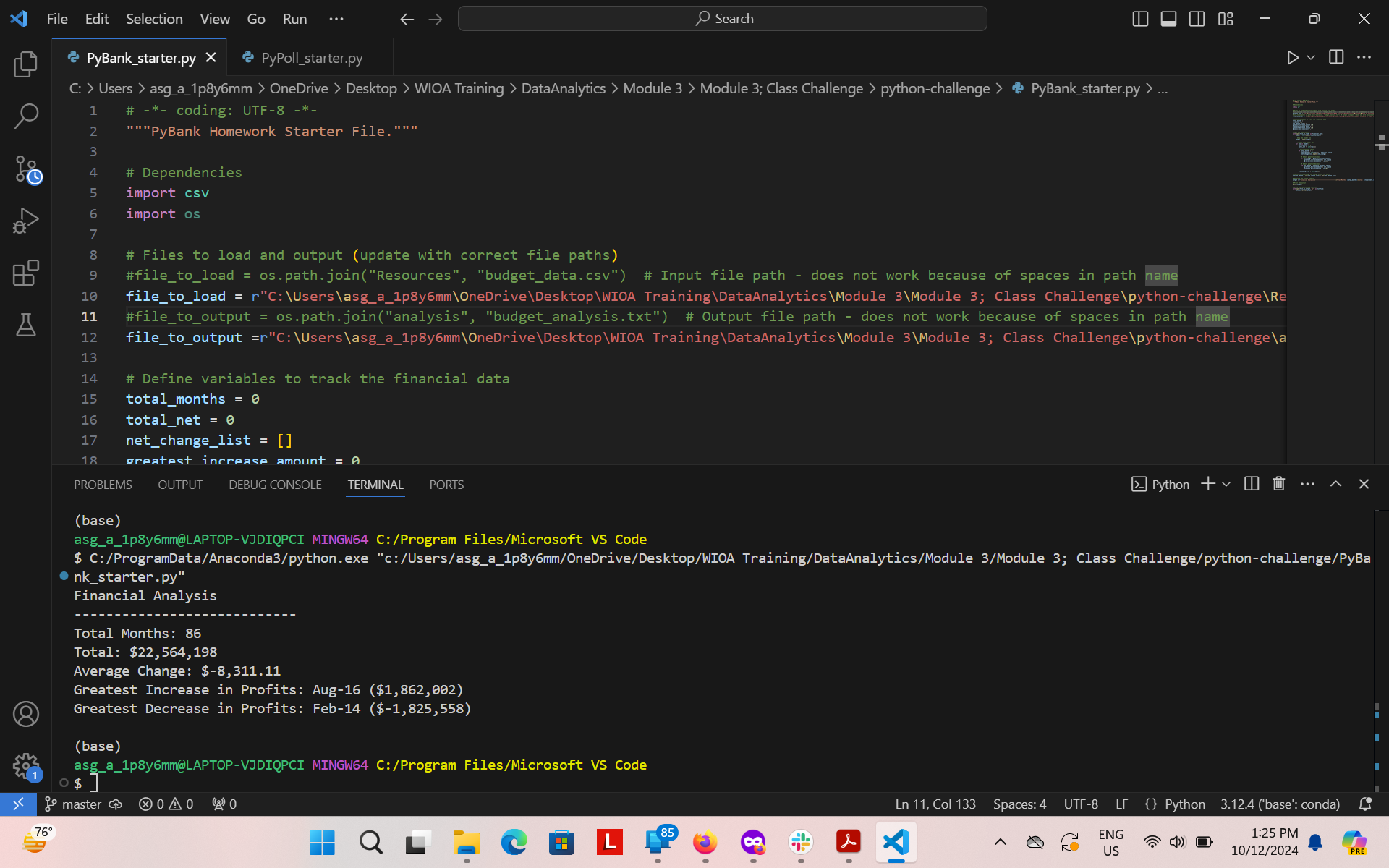
**PyBank Starter**

**Explanation:**

1. **Import Libraries:**
   * csv: For reading and processing the CSV file.
   * os: For handling file paths.
2. **File Paths:**
   * Set file\_to\_load to the path of your CSV file.
   * Set file\_to\_output to the desired path for the output text file.
3. **Variables:**
   * total\_months: Stores the total number of months.
   * total\_net: Stores the total profit/loss over all months.
   * net\_change\_list: A list to store the changes in profit/loss between consecutive months.
   * greatest\_increase\_amount: Stores the maximum profit increase amount.
   * greatest\_increase\_month: Stores the month of the maximum profit increase.
   * greatest\_decrease\_amount: Stores the maximum profit decrease amount.
   * greatest\_decrease\_month: Stores the month of the maximum profit decrease.
4. **Open and Read CSV:**
   * Open the CSV file using with open(file\_to\_load) as financial\_data:.
   * Create a csv.reader object to read the data row by row.
   * Skip the header row using header = next(reader).
5. **Process Data:**
   * Iterate through each row in the CSV using for row in reader:.
   * Increment total\_months.
   * Add the current month's profit/loss to total\_net.
   * Calculate the net change between the current month and the previous month (if it's not the first month).
   * Append the net change to net\_change\_list.
   * Update greatest\_increase\_amount and greatest\_increase\_month if the current net change is greater.
   * Update greatest\_decrease\_amount and greatest\_decrease\_month if the current net change is less.
6. **Calculate Average Change:**
   * Calculate the average net change using sum(net\_change\_list) / len(net\_change\_list).
7. **Generate Output:**
   * Create a formatted string output containing all the calculated values.
8. **Print and Write to File:**
   * Print the output to the terminal using print(output).
   * Open the output text file (file\_to\_output) in write mode ("w") and write the output to it.
   * Requested output



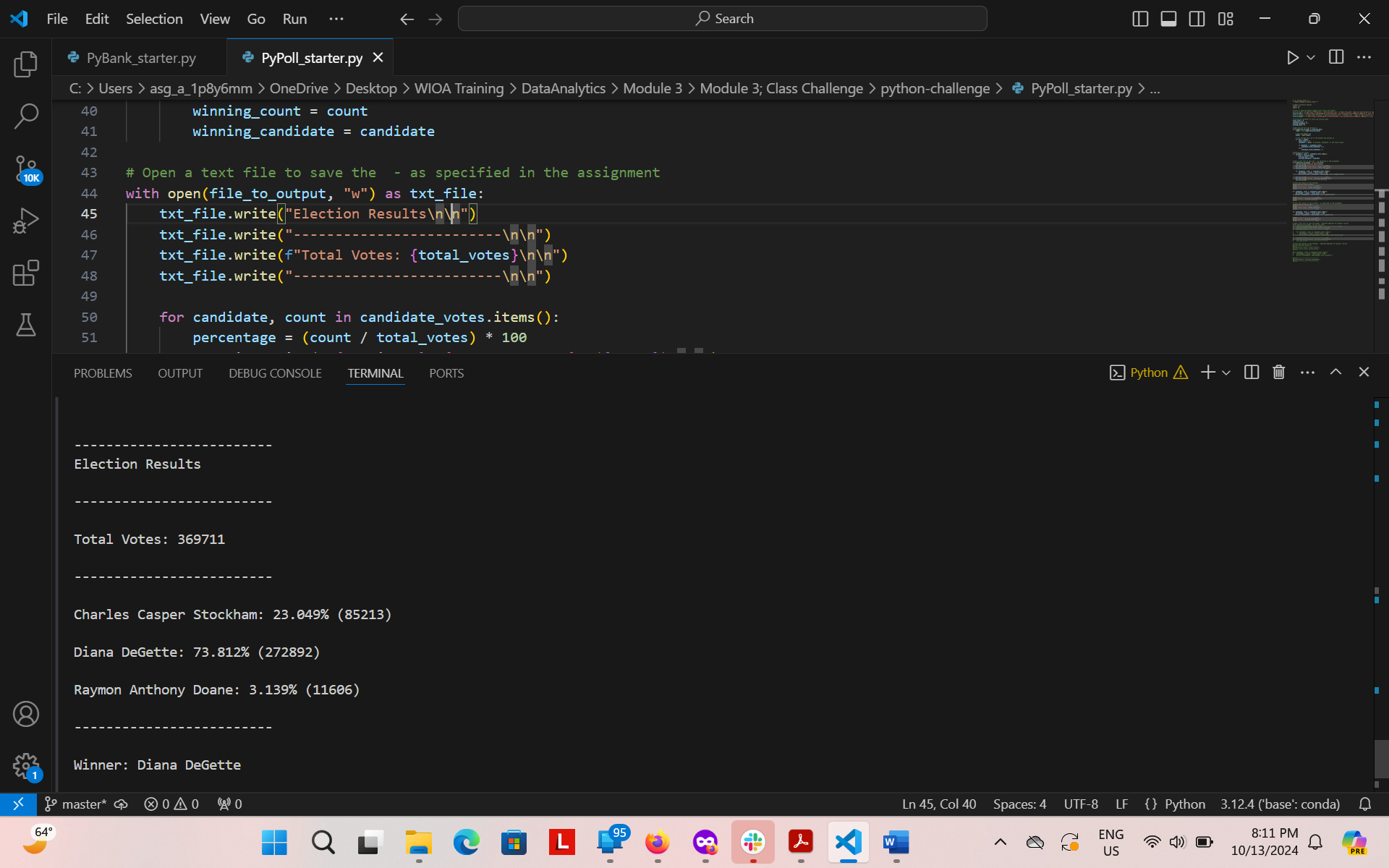
* + Improved output



**PyPoll Starter**

**Explanation:**

1. **Import Modules:** Import csv for reading the CSV file and os for file path handling.
2. **File Paths:** Set the input and output file paths.
3. **Initialize Variables:**
   * total\_votes: Stores the total number of votes.
   * candidate\_votes: A dictionary to store candidate names as keys and their vote counts as values.
   * winning\_candidate: Stores the name of the winning candidate.
   * winning\_count: Stores the number of votes the winner received.
4. **Open and Read CSV:**
   * Open the CSV file using with open(file\_to\_load) as election\_data:.
   * Create a csv.reader object to read the data row by row.
   * Skip the header row using header = next(reader).
5. **Process Data:**
   * Loop through each row in the CSV using for row in reader:.
   * Increment total\_votes for each row.
   * Extract the candidate's name from the third column (row[2]).
   * Update the candidate\_votes dictionary:
     + If the candidate is already in the dictionary, increment their vote count.
     + If the candidate is new, add them to the dictionary with a vote count of 1.
6. **Determine the Winner:**
   * Loop through the candidate\_votes dictionary.
   * For each candidate, compare their vote count to winning\_count.
   * If a candidate has more votes than the current winning\_count, update winning\_candidate and winning\_count.
7. **Write Output to File:**
   * Open the output text file using with open(file\_to\_output, "w") as txt\_file:.
   * Write the election results to the file, including:
     + Total votes
     + Each candidate's name, vote count, and percentage
     + The winner
8. **Print Output to Terminal:**
   * Print the election results to the terminal using print() statements.
   * Output according to assignment specifications:



* Output modified for greater output clarity