**AC50002**

**PROGRAMMING LANGUAGES FOR DATA ENGINEERING**

**PYTHON ASSIGNMENT**

The code, input files and output file: <https://github.com/Appukgowda/pythonassig.git>

**EXPLANATION OF CODE-**

**Load Values from file:**

import string

def load\_values(file\_path):

values = {}

with open(file\_path, 'r') as file:

for line in file:

try:

letter, value = line.strip().split()

values[letter] = int(value)

except ValueError:

pass

return values

* Importing a ‘string’ module to use its ‘punctuation’ constant.
* Defining a function ‘load\_values’ that reads values from a file into a dictionary. This file (‘values.txt’) contains scores for each letter.
* The function handles line that lack the proper format by using a ‘try-except’ block.

**Calculate score function:**

def calculate\_score(letter, position, is\_first, is\_last, values):

try:

if is\_first:

return 0

elif is\_last:

return 20 if letter == 'E' else 5

else:

position\_value = 1 if position == 2 else (2 if position == 3 else 3)

return position\_value + values.get(letter, 0)

except KeyError:

return 0

return values

* A function ‘calculate\_score’ that calculates a letter’s score based on its position and context in a word.
* It considers whether the letter is the first or last in a word and uses a dictionary (‘values’) to look up the score for the letter.
* Handling scenarios where the letter is missing from the values using a ‘try-except’ block.

**Generate Abbreviations function:**

def generate\_abbreviations(name, values):

words = [word.strip(string.punctuation) for word in name.split()]

abbreviations = set()

for word in words:

first\_letter = word[0]

for i in range(1, len(word) - 1):

second\_letter = word[i]

for j in range(i + 1, len(word)):

third\_letter = word[j]

abbreviation = f"{first\_letter}{second\_letter}{third\_letter}"

score = calculate\_score(second\_letter, i, False, j == len(word) - 1, values) + calculate\_score(third\_letter, j, False, True, values)

abbreviations.add((abbreviation, score))

return abbreviations

* A function ‘generate\_abbreviations’ that takes a name and values dictionary as input.
* It splits the name into words, removes punctuation, and generates three-letter abbreviations for each word.
* The function uses the ‘calculate\_score’ function to calculate the score for each abbreviation.

**Main Function:**

def main():

input\_file = input("Enter the name of the input file (with .txt extension): ")

values\_file = "values.txt" # You may need to change this based on your file location

values = load\_values(values\_file)

with open(input\_file, 'r') as file:

names = [line.strip() for line in file]

surname = input("Enter your surname: ")

output\_file = f"{surname.lower()}\_{input\_file[:-4]}\_abbrevs.txt"

with open(output\_file, 'w') as file:

for name in names:

abbreviations = generate\_abbreviations(name.upper(), values)

best\_abbreviation, best\_score = min(abbreviations, key=lambda x: x[1])

file.write(f"{name.upper()}: {best\_abbreviation} ({best\_score} points)\n")

* The ‘main’ function is the core of the script. It interacts with the user, loads values, reads names from the input file, and generate abbreviations.
* It prompts the user for input file names, including the surname for the output file.
* For each name, it generates abbreviations, selects the one with the lowest score and writes the results to the output file.

**Script Execution:**

if \_name\_ == "\_main\_":

    main()

* The script checks if it is being run as the main program (‘if\_name\_==”\_main\_”) and if so, executes the ‘main’ function.









