Anurat Wongbunmak
2488941

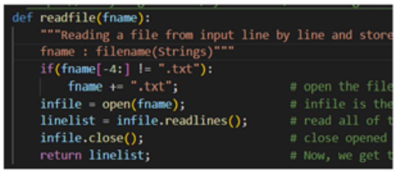
AC50002 - Programming languages for Data Engineering (SEM 1 22/23)

Python Assignment

This link will take you to my GitHub repository, where you can access the code for this assignment.

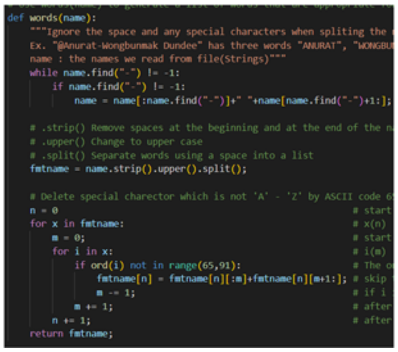
https://github.com/EarnAnurat/PythonAsgmt1_Anurat2488941

The code is made up of five functions that work together to complete the task including readfile(fname), words(name), allabbr(wl) ,cal(wl,abbr) and main().



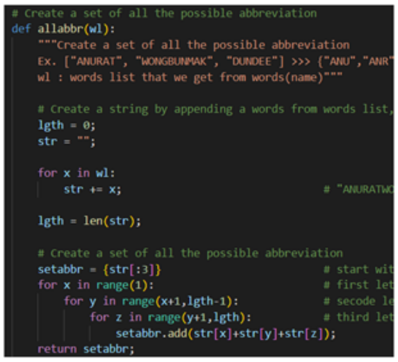
```python
def readfile(fname):
    """Reading a file from input line by line and store
    fname : filename(Strings)"""
    if(fname[-4:] != ".txt"):
        fname += ".txt";                        # open the file
    infile = open(fname);                       # infile is the
    linelist = infile.readlines();              # read all of t
    infile.close();                             # close opened
    return linelist;                            # Now, we get t
```

First, the **readfile()** function reads every line from the file into a list using the .readlines() method, where each line is an item in the list. As a result, it then returns the data "linelist", which is a list of every line.



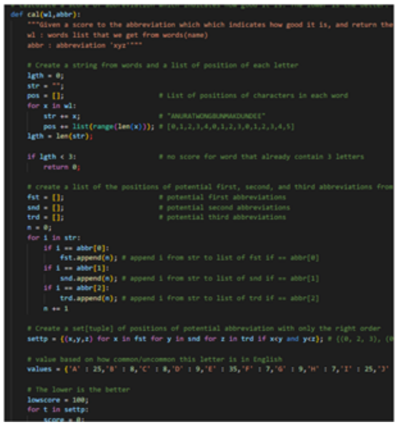
```python
def words(name):
    """Ignore the space and any special characters when spliting the
    Ex. "@Anurat-Wongbunmak Dundee" has three words "ANURAT", "WONGBU
    name : the names we read from file(Strings)"""
    while name.find("-") != -1:
        if name.find("-") != -1:
            name = name[:name.find("-")]+" "+name[name.find("-")+1:];

    # .strip() Remove spaces at the beginning and at the end of the n
    # .upper() Change to upper case
    # .split() Separate words using a space into a list
    fmtname = name.strip().upper().split();

    # Delete special charector which is not 'A' - 'Z' by ASCII code 6
    n = 0                                                         # start
    for x in fmtname:                                             # x(n)
        m = 0;                                                    # start
        for i in x:                                               # i(m)
            if ord(i) not in range(65,91):                        # The o
                fmtname[n] = fmtname[n][:m]+fmtname[n][m+1:]; # skip
                m -= 1;                                           # if i
            m += 1;                                               # after
        n += 1;                                                   # after
    return fmtname;
```

Second, the **words()** function will get the stored data called “linelist” and turn it into a list of words that can be used as abbreviations. Replace "-" with a space to start, then use.strip() to eliminate the space between them. .split() is used to break apart words, and.upper() is used to change to uppercase. Last step, deleting all special characters with ASCII codes 65-90 that are letters "A" through "Z"



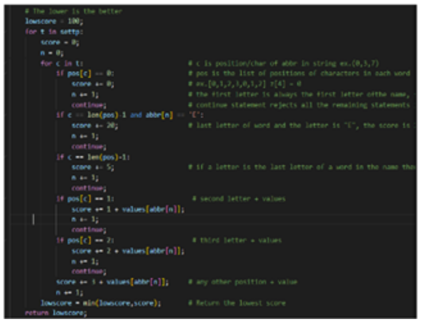
```python
# Create a set of all the possible abbreviation
def allabbr(wl):
    """Create a set of all the possible abbreviation
    Ex. ["ANURAT", "WONGBUNMAK", "DUNDEE"] >>> {"ANU","ANR"
    wl : words list that we get from words(name)"""

    # Create a string by appending a words from words list,
    lgth = 0;
    str = "";

    for x in wl:
        str += x;                                   # "ANURATWO

    lgth = len(str);

    # Create a set of all the possible abbreviation
    setabbr = {str[:3]}                             # start wit
    for x in range(1):                              # first let
        for y in range(x+1,lgth-1):                 # secode le
            for z in range(y+1,lgth):               # third let
                setabbr.add(str[x]+str[y]+str[z]);
    return setabbr;
```

Third, to construct a collection of all potential abbreviations, the **allabbr()** function takes a list of words that are appropriate for abbreviations that we obtain from the previous function. The initial letter, is always be the range(1) which is the first letter of words. The second letters, range(x+1, lgth-1), are not the first and last letters of words. The third letter, range(y+1, lgth), will be one that comes from the rest of the characters that come after the second.

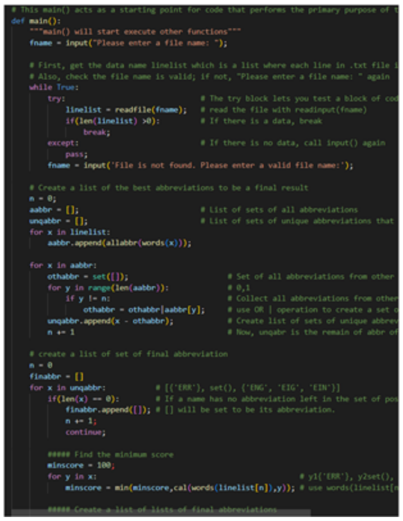


```python
def cal(wl,abbr):
    """Given a score to the abbreviation which which indicates how good it is, and return the
    wl : words list that we get from words(name)
    abbr : abbreviation 'xyz'"""

    # Create a string from words and a list of position of each letter
    lgth = 0;
    str = "";
    pos = [];                       # List of positions of characters in each word
    for x in wl:
        str += x;                   # "ANURATWONGBUNMAKDUNDEE"
        pos += list(range(len(x))); # [0,1,2,3,4,0,1,2,3,0,1,2,3,4,5]
    lgth = len(str);

    if lgth < 3:                    # no score for word that already contain 3 letters
        return 0;

    # create a list of the positions of potential first, second, and third abbreviations from
    fst = [];                       # potential first abbreviations
    snd = [];                       # potential second abbreviations
    trd = [];                       # potential third abbreviations
    n = 0;
    for i in str:
        if i == abbr[0]:
            fst.append(n); # append i from str to list of fst if == abbr[0]
        if i == abbr[1]:
            snd.append(n); # append i from str to list of snd if == abbr[1]
        if i == abbr[2]:
            trd.append(n); # append i from str to list of trd if == abbr[2]
        n += 1

    # Create a set[tuple] of positions of potential abbreviation with only the right order
    settp = {(x,y,z) for x in fst for y in snd for z in trd if x<y and y<z}; # {(0, 2, 3), (0

    # value based on how common/uncommon this letter is in English
    values = {'A' : 25,'B' : 8,'C' : 8,'D' : 9,'E' : 35,'F' : 7,'G' : 9,'H' : 7,'I' : 25,'J'

    # The lower is the better
    lowscore = 100;
    for t in settp:
        score = 0;
```

Fourth, the **cal()** function gives the abbreviation a score that represents its effectiveness and then returns the score with the lowest number. An abbreviation's overall score is the result of adding the scores for its second and third letters. These individual letter scores are based on the letter's position in the word and how common or uncommon it is in English.



```python
    # The lower is the better
    lowscore = 100;
    for t in settp:
        score = 0;
        n = 0;
        for c in t:                         # c is position/char of abbr in string ex.(0,3,7)
            if pos[c] == 0:                 # pos is the list of positions of characters in each word
                score += 0;                 # ex.[0,1,2,3,0,1,2] p[4] = 0
                n += 1;                     # the first letter is always the first letter ofthe name,
                continue;                   # continue statement rejects all the remaining statements
            if c == len(pos)-1 and abbr[n] == "E":
                score += 20;                # last letter of word and the letter is "E", the score is
                n += 1;
                continue;
            if c == len(pos)-1:
                score += 5;                 # if a letter is the last letter of a word in the name tho
                n += 1;
                continue;
            if pos[c] == 1:                  # second letter + values
                score += 1 + values[abbr[n]];
                n += 1;
                continue;
            if pos[c] == 2:                  # third letter + values
                score += 2 + values[abbr[n]];
                n += 1;
                continue;
            score += 3 + values[abbr[n]];   # any other position + value
            n += 1;
        lowscore = min(lowscore,score);     # Return the lowest score
    return lowscore;
```



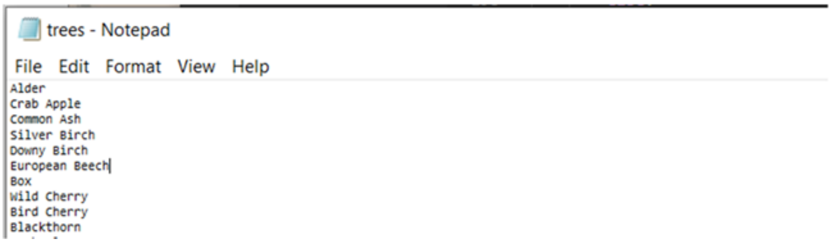
```
# This main() acts as a starting point for code that performs the primary purpose of t
def main():
    """main() will start execute other functions"""
    fname = input("Please enter a file name: ");

    # First, get the data name linelist which is a list where each line in .txt file i
    # Also, check the file name is valid; if not, "Please enter a file name: " again
    while True:
        try:                                # The try block lets you test a block of cod
            linelist = readfile(fname);     # read the file with readinput(fname)
            if(len(linelist) >0):           # If there is a data, break
                break;
        except:                             # If there is no data, call input() again
            pass;
        fname = input('File is not found. Please enter a valid file name:');

    # Create a list of the best abbreviations to be a final result
    n = 0;
    aabbr = [];                             # List of sets of all abbreviations
    unqabbr = [];                           # List of sets of unique abbreviations that
    for x in linelist:
        aabbr.append(allabbr(words(x)));

    for x in aabbr:
        othabbr = set([]);                       # Set of all abbreviations from other
        for y in range(len(aabbr)):              # 0,1
            if y != n:                           # Collect all abbreviations from other
                othabbr = othabbr|aabbr[y];      # use OR | operation to create a set o
        unqabbr.append(x - othabbr);             # Create list of sets of unique abbrev
        n += 1                                   # Now, unqabr is the remain of abbr of

    # create a list of set of final abbreviation
    n = 0
    finabbr = []
    for x in unqabbr:               # [{'ERR'}, set(), {'ENG', 'EIG', 'EIN'}]
        if(len(x) == 0):            # If a name has no abbreviation left in the set of pos
            finabbr.append([]); # [] will be set to be its abbreviation.
            n += 1;
            continue;

        ##### Find the minimum score
        minscore = 100;
        for y in x:                                               # y1{'ERR'}, y2set(),
            minscore = min(minscore,cal(words(linelist[n]),y)); # use words(linelist[n

        ##### Create a list of lists of final abbreviations
```

Finally, **main()** function serves as a starting point for code that performs the primary purpose of the script. It will start execute other functions. Use **readfile()** to get the data name linelist which is a list where each line in .txt file is an item. Then create a list of the best abbreviations to be a final result with the help of **words(), allabbr()** and **cal()**. And finally write the output file.

if __name__ == "__main__": is to ensure that when the module is simply imported, the function is not called, Then if the modules are the main, the module can be run from an ordinary Windows command prompt.

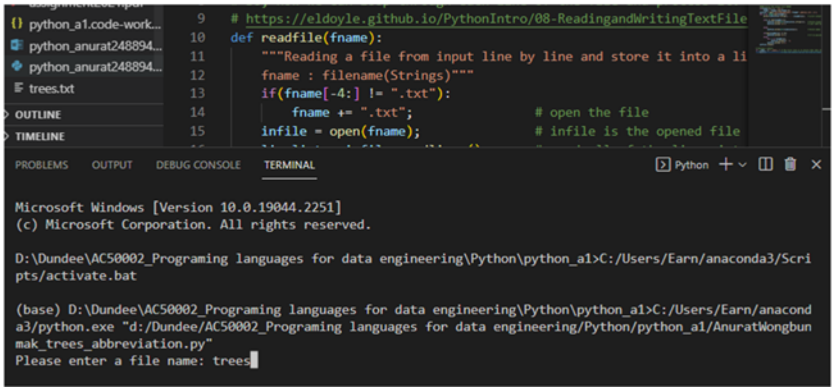
Using the modules

Input file



```
Alder
Crab Apple
Common Ash
Silver Birch
Downy Birch
European Beech
Box
Wild Cherry
Bird Cherry
Blackthorn
```

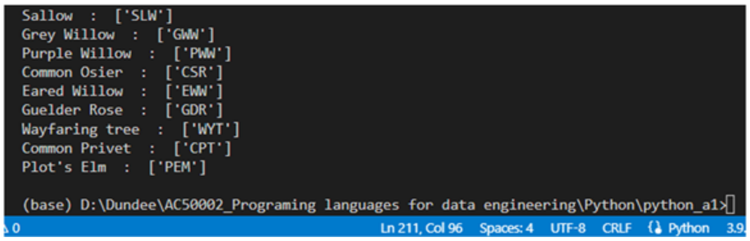
enter a file name: trees



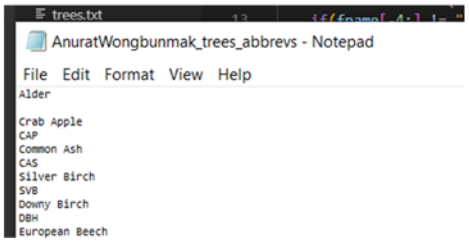
```
# https://eldoyle.github.io/PythonIntro/08-ReadingandWritingTextFile
def readfile(fname):
    """Reading a file from input line by line and store it into a li
    fname : filename(Strings)"""
    if(fname[-4:] != ".txt"):
        fname += ".txt";                    # open the file
    infile = open(fname);                   # infile is the opened file
```

```
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

D:\Dundee\AC50002_Programing languages for data engineering\Python\python_a1>C:/Users/Earn/anaconda3/Scri
pts/activate.bat

(base) D:\Dundee\AC50002_Programing languages for data engineering\Python\python_a1>C:/Users/Earn/anacond
a3/python.exe "d:/Dundee/AC50002_Programing languages for data engineering/Python/python_a1/AnuratWongbun
mak_trees_abbreviation.py"
Please enter a file name: trees
```

Output file



```
Sallow  :  ['SLW']
Grey Willow  :  ['GWW']
Purple Willow  :  ['PWW']
Common Osier  :  ['CSR']
Eared Willow  :  ['EWW']
Guelder Rose  :  ['GDR']
Wayfaring tree  :  ['WYT']
Common Privet  :  ['CPT']
Plot's Elm  :  ['PEM']

(base) D:\Dundee\AC50002_Programing languages for data engineering\Python\python_a1>
```



trees.txt
AnuratWongbunmak_trees_abbrevs - Notepad
File Edit Format View Help

```
Alder

Crab Apple
CAP
Common Ash
CAS
Silver Birch
SVB
Downy Birch
DBH
European Beech
```