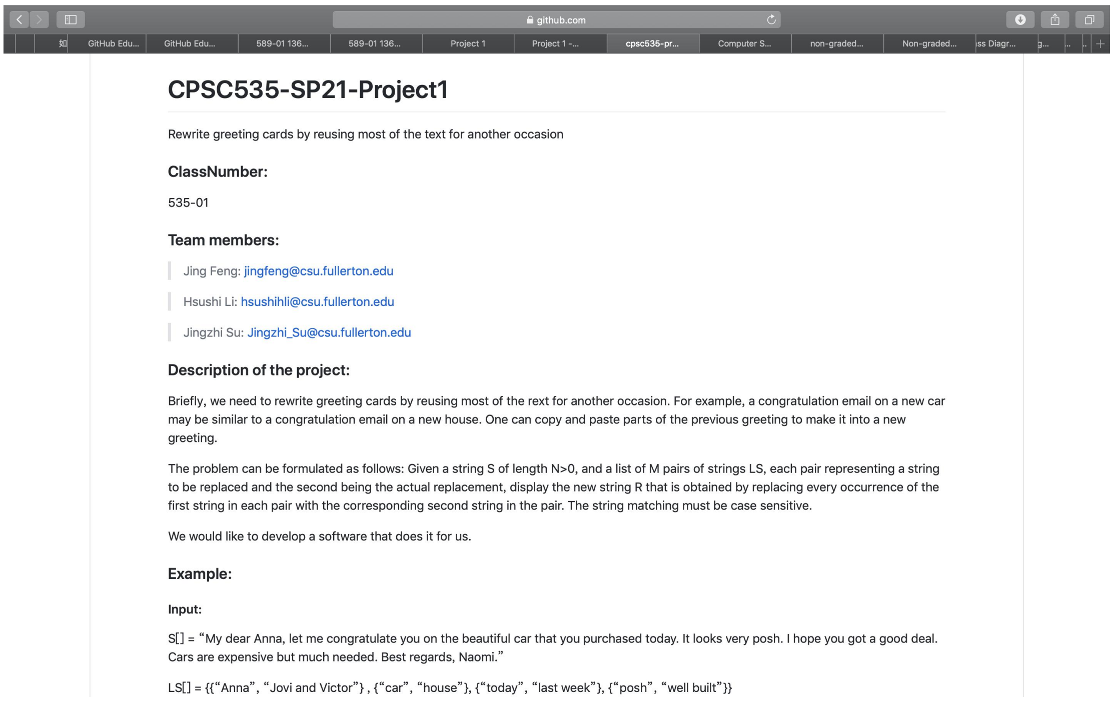
CPSC535-SP21-Project1

Team members:

Jing Feng: jingfeng@csu.fullerton.edu

Hsushi Li: hsushihli@csu.fullerton.edu

Jingzhi Su: Jingzhi_Su@csu.fullerton.edu



CPSC535-SP21-Project1

Rewrite greeting cards by reusing most of the text for another occasion

ClassNumber:

535-01

Team members:

> Jing Feng: jingfeng@csu.fullerton.edu
>
> Hsushi Li: hsushihli@csu.fullerton.edu
>
> Jingzhi Su: Jingzhi_Su@csu.fullerton.edu

Description of the project:

Briefly, we need to rewrite greeting cards by reusing most of the rext for another occasion. For example, a congratulation email on a new car may be similar to a congratulation email on a new house. One can copy and paste parts of the previous greeting to make it into a new greeting.

The problem can be formulated as follows: Given a string S of length N>0, and a list of M pairs of strings LS, each pair representing a string to be replaced and the second being the actual replacement, display the new string R that is obtained by replacing every occurrence of the first string in each pair with the corresponding second string in the pair. The string matching must be case sensitive.

We would like to develop a software that does it for us.

Example:

Input:

S[] = "My dear Anna, let me congratulate you on the beautiful car that you purchased today. It looks very posh. I hope you got a good deal. Cars are expensive but much needed. Best regards, Naomi."

LS[] = {{"Anna", "Jovi and Victor"} , {"car", "house"}, {"today", "last week"}, {"posh", "well built"}}

Pseudocode

```
def replace (S, dict):
    result = “” # indicates the new string after replacement and will return at the end
    left = 0       # the leftmost position when checking the undone part of old S
    right = 0      # the rightmost position when checking the undone part of old S
    cur = 0         # the current position when checking the undone part of old S
    hasMatch = True    # a flag to indicate if the old substring matches

    while cur < S.length: # while current point is in the boundary of the length of S, do
        cur_char = S[cur]
        If cur_char in the dictionary
            right = cur
            for each pair in the entry of dictionary[cur_char]
                for j in the range of [0, the length of the old substring in each pair)
                    if a certain char of the old substring doesn't match
                        reset hasMatch = false
                        break the inner for loop to check the next pair in this entry if it exists
                    right += 1
                If the entire substring match
                    result = the previous part of S and new substring
                    cur = the first positon of the old substring + the length of the old substring
                    left = cur
                    break
            If no substring match
                reset hasMatch flag = True
                cur += 1
        else
            cur += 1

    if left <    S.length    # check if the last part of old S has added to the result
        result += last part
    return result


if __name__ == “__main__”:
    filename_S = the user input S
    filename_LS = the user input LS
    LS = []
    try:
        open filename_LS as f_LS:
            for each line in f_LS
```

```
                read LS file line by line
                convert the string LS to list LS

            dict = {}, a dictionary to restore the LS list
            for each pair in LS:
            If each[0][0] not in the dictionary
                dict[each[0][0]] = [each]
            else
                dict[each[0][0]].append(each)

        open filename_S as f_S
            old_S = “”
            new_S = “”
            for each line in f_S
                call replace() function and get the new S
            print the new S to screen

    except Exception as reason: #if there is no such file or the file cannot be opened, an error is
raised
        print(reason)
```

Edge Cases:

Here, two edge cases are specifically explained. We hope the approach we take could meet the project requirements.

(1) Two old substrings in the LS list overlap each other.

```
For example:
S[]= "My teacher told me to remember that today is a new day. The old principal."
LS[] = {{"to", "yester"},{"old", "new"}}
```

In the above example, the "to" and "old" overlap in the word "told".
Because there is no specific requirement in the project document about this case, the strategy of our algorithm is that if we encounter words like "told", we only replace "to" with "yester", not "old" with "new", because we scan from left to right.

(2) One old substring in the LS list contains another old substring in LS

```
For example:
S[]= "My teacher told me to remember that today is a new day. The old principal."
LS[] = {{"to", "yester"},{"today", "tomorrow"}}
```

In the above example, the word "today" contains "to".
Because there is no specific requirement in the project document about this case, the strategy of our algorithm is that if we encounter words like "today", according to the order of "to" and "today" in the LS list user gave us, we only replace the previous one. In this example, since "to" comes before "today", so we only replace "to" with "yester"

Description of how to run the code:

1. Download the program rewriting_greeting_ecards.py to your computer
2. Open the terminal and change the current working directory
3. Run the executable file in the terminal (Please use python3 to run the program)

```
python3 rewriting_greeting_ecards.py
```

4. Enter two .txt files names you wish to apply rewriting.

```
If your test code is in the current working directory, you can type the filename directly.
For example:
     S.txt
     LS.txt
If not, you need to type the absolute pathname of the file into the terminal and return.
For example:
     /Users/Jim/Desktop/S.txt
     /Users/Jim/Desktop/LS.txt
```

5. After running the program, it will print the output to the screen.

*** About the format of user input files:**

(1) The content in the S.txt should look like as follows (for example 1). There is no "" at the beginning and end of the entire string.

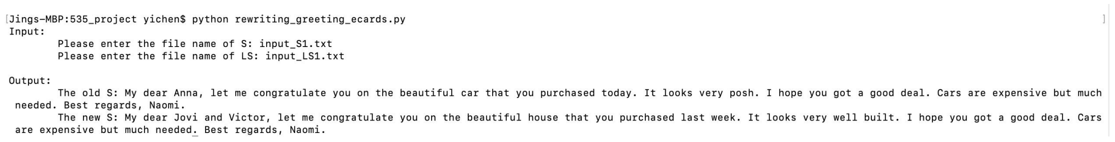
```
My dear Anna, let me congratulate you on the beautiful car that you purchased today. It looks very posh. I hope you got a good deal. Cars are expensive but much needed. Best regards, Naomi.
```

(2) The content in the LS.txt should look like as follows (for example 1). There are {} at the beginning and end of the entire list, and each pair inside is also surrounded by {}. The old substring and the new substring in each pair need to be surrounded by "".

```
{{“Anna”, “Jovi and Victor”},{“car”, “house”},{“today”, “last week”},{“posh”, “well built”}}
```

Three snapshots of code executing

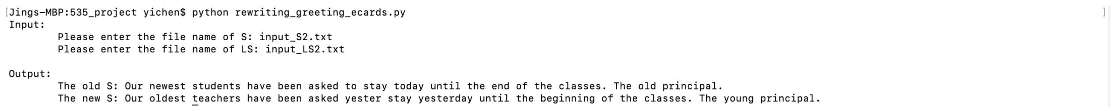
Example1



```
[Jings-MBP:535_project yichen$ python rewriting_greeting_ecards.py                                                                                    ]
 Input:
        Please enter the file name of S: input_S1.txt
        Please enter the file name of LS: input_LS1.txt

 Output:
        The old S: My dear Anna, let me congratulate you on the beautiful car that you purchased today. It looks very posh. I hope you got a good deal. Cars are expensive but much
  needed. Best regards, Naomi.
        The new S: My dear Jovi and Victor, let me congratulate you on the beautiful house that you purchased last week. It looks very well built. I hope you got a good deal. Cars
  are expensive but much needed. Best regards, Naomi.
```

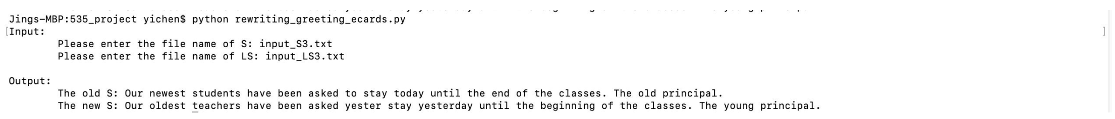
Example2



```
[Jings-MBP:535_project yichen$ python rewriting_greeting_ecards.py                                                                                    ]
 Input:
        Please enter the file name of S: input_S2.txt
        Please enter the file name of LS: input_LS2.txt

 Output:
        The old S: Our newest students have been asked to stay today until the end of the classes. The old principal.
        The new S: Our oldest teachers have been asked yester stay yesterday until the beginning of the classes. The young principal.
```

Example3



```
 Jings-MBP:535_project yichen$ python rewriting_greeting_ecards.py
[Input:                                                                                                                                               ]
        Please enter the file name of S: input_S3.txt
        Please enter the file name of LS: input_LS3.txt

 Output:
        The old S: Our newest students have been asked to stay today until the end of the classes. The old principal.
        The new S: Our oldest teachers have been asked yester stay yesterday until the beginning of the classes. The young principal.
```