Report of assignment2

Jensen D1265209

```c
int main(){
    char *buffer;
    int capacity = 512;
    FILE *dataIn, *dataOut;
    int length = 0;
    buffer = (char*)malloc(capacity*sizeof(char));
    char *vowels = "AEIOU";
    int c;

    dataIn = fopen("Gift_of_Magi.txt", "r");
    if(dataIn == NULL){
        printf("\nText file \"Gift_of_Magi.txt\" does not exist.\n\n");
        return -1;
    }
```

First, I defined the variables I need, including the space of memory, the word of vowels. Of course, some of them were added after, but I explain here together. Also open the file, txt document. And if the opening failed it will print out some words.

```c
    while ((c = fgetc(dataIn)) != EOF) {
        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z')) {
            c = toupper(c);
            if (length == capacity) {
                capacity += 512;
                buffer = (char*)realloc(buffer, capacity * sizeof(char));
                if (buffer == NULL) {
                    printf("Memory reallocation failed\n");
                    fclose(dataIn);
                    return -1;
                }
            }
            buffer[length++] = c;
        }
        else
            printf("input character is not an English letter");
    }
    buffer[length] = '\0';
    printf(">>>> Total input English characters: %d\n",length);
```

And then, make sure all the word is English letter, from a to z in small letters or capital letters. If it is, then, make transfer the letter to capital letter, and put the letter into the space of buffer. At the same time, make sure the space of memory is enough to store the letters. If not enough, add more space which 512 in once. Therefore, have to realloc the memory.

```
dataOut = fopen("result.txt", "w");
fwrite(buffer, sizeof(char), length, dataOut);
fclose(dataIn);
```

Output the other file which is all the capital letters, and no space inside the string.

```
printf(">>>The first 800 characters are: \n");
int i;
printf("  ");
for (i = 0; i < 800; i++) {
    printf("%c", buffer[i]);
    if ((i + 1) % 80 == 0) {
        printf("\n  ");
    }
}
```

Print out first 800 characters. I used for loop to print it out from the fist letter which is stored on buffer[0]. Because each line can only have 80 words, there is if loop to make sure it will have new line in each 80 words.

```
printf(">>>>The number of contiguous letter(s) are: \n");
int one_char = 0, two_char = 0, three_char = 0, four_or_more_char = 0;
int current_len = 0;
for(i=0; i<length; i++){
    current_len++;
    if(buffer[i] != buffer[i+1]){
        if(current_len == 1)
            one_char++;
        else if(current_len == 2)
            two_char++;
        else if(current_len == 3)
            three_char++;
        else
            four_or_more_char++;
        current_len = 0;
    }
}
printf("  One character: %d\n", one_char);
printf("  Two contiguous character: %d\n", two_char);
printf("  Three contiguous character: %d\n", three_char);
printf("  Four or more contiguous character: %d\n", four_or_more_char);
printf("  ****Total characater counts: %d\n", length);
```

To count the contiguous letters. There is another variables, current_len, used to count the contiguous times. In this for loop, in the beginning, current_len will plus one, and go into if loop, if next letter are the same letter, then go for for loop again. Until the letter is not the same as the letter before. So, because of the current_len number, can find that how many times it continues. After that, just print out the number.

```c
int count_A = 0, count_E = 0, count_I = 0, count_O = 0, count_U = 0;
int total_vowels = 0;
char *current_pos = buffer;
while ((current_pos = strpbrk(current_pos, vowels)) != NULL) {
switch (*current_pos) {
    case 'A': count_A++; break;
    case 'E': count_E++; break;
    case 'I': count_I++; break;
    case 'O': count_O++; break;
    case 'U': count_U++; break;
    }
current_pos++;
}
total_vowels = count_A + count_E + count_I + count_O + count_U;
printf(">>>> The number of occurences of vowels: \n");
printf("  Vowels 'A': %d\n", count_A);
printf("  Vowels 'E': %d\n", count_E);
printf("  Vowels 'I': %d\n", count_I);
printf("  Vowels 'O': %d\n", count_O);
printf("  Vowels 'U': %d\n", count_U);
printf("  Total number of vowels: %d\n", total_vowels);
```

Here is to count the number of letters, AEIOU. And I used pointer here. The pointer begins from the beginning of buffer. And go into the while loop. The new pointer will at the same letter which strpbrk() find. And used switch to count the number. Until strpbrk cant find the same letter anymore. Finally, printout.