LAB [1], [01/10/2019] **MCS 253P**

Name: Yu Qin
Partner: Yuwen Suo

Problem Description
Read numbers from a file, To get the average value of those numbers.
!: No dynamic allocation.

Additional Problem Specifics
Input file type: file.dat (and I later treat the file without the suffix, but no binary file.)
Number value overflow? Valid number?: Yes, number
Integer: No, floating numbers
Negative: Yes, maybe
Output format: numbers scale, avg number
Empty? : Given No empty, and I later give information for users.
too many input numbers? It has not too much differences between one input number and one million inputer number, since the process is designed to scan the file line by line.
too larger input numbers? Double is much enough; if really more larger, it is another trivial point.

Sample Input
1st) 1, 3 ------- 2
2nd) 10,20,30 ------- 20
3rd) -1 ----- - 1
4th) empty file
5th) different file name and suffix (eg., file.dat, file.txt, etc.)

Proposed Algorithm

Description:
The overall thought is to calculate the sum and the number count, and get the quotient
So, maintain a double for the sum, and a double for the count.
All the data should be read from the file.

Correctness:
It is based on the math concept. It works for both positive numbers and negative numbers.

Time Complexity:
It scans the file while update the sum and count, so it's O(n).

Space Complexity:
It just need a double number for sum and count, so it's O(1).

C++ Implementation of Algorithm
```
#include <iostream>
#include <fstream>
using namespace std;
```

```
int main(int argc, char** argv){
        //open file
        if (argc < 2){
                cout<<"You should input the file name.\n\n";
                return -0;
        }
        ifstream infile(argv[1]);
        if ( !infile.is_open() ){
                cout<<"Cannot open the file.\n\n";
                return -0;
        }

        //read file
        double int_temp;
        double sum = 0, count = 0, average = 0;
        while ( infile >> int_temp) {
                sum = sum + int_temp;
                count += 1;
        }
        infile.close();

        //calculate
        if (count == 0 ){
                cout<<"Your input file is empty.\n\n";
                return -0;
        } else{
                average = sum/count;
                cout<<"The average of the "<< count <<" numbers in file '" << argv[1] << "' is "<< average <<
endl<<endl;
                return 0;
        }
}
```

Edge Test Cases

- sample description of an edge test case
    - output we expect (want)
    - output our algorithm produces
- no file name input in command

    expected: remind of "You should input the file name" and exit.
    output our algorithm produces: remind of "You should input the file name" and exit.
    conclusion: right 🆗
- file open fail (maybe due to wrong file name, lack of permission, etc.)

    expected: remind of "Cannot open the file." and exit.
    output our algorithm produces: remind of "Cannot open the file." and exit.
    conclusion: right 🆗
- empty file

    expected: remind of "Your input file is empty." and exit.
    output our algorithm produces: remind of "Your input file is empty." and exit.
    conclusion: right 🆗
- corner case of data are considered in former "Additional Problem Specifics" part.

Comments

Good question, basic and realistic to be used.
Be cautious to the input file processing.
Sometime we may need to handle the valid number input.