CS 2400 Credit Card Balances Due: October 22, 2019

40 Points

A credit card company keeps track of its customers' data in a text file. The file is supplied by another department within the company and is generated every day. The generated file includes a customer ID, first name, last name, current balance, and the number of days before the balance is due. The bank would like to generate reports about the customers stored in this daily file. For example, how many customers have a due date within the next five days?

Write a C++ program to process the data file. Each line in the file contains the following data and each item is separated by at least one space:

```
<id> <first> <last> <balance> <days to due date>
```

Examples:

```
     12340 Jimmy Carter 3500.23 4
     22345 Richard Nixon 5600.21 0
```

The program should process the file and output all the customers that match a given criteria into an output file. The criteria will be specified using command line arguments as follows:

`-a <balance> <input file> <output file>`

All the customers with a balance equal or higher than a specified balance
Example: `./a.out -a 2000 data.txt output.txt`

`-b <balance> <input file> <output file>`

All the customers with a balance equal or below the specified balance
Example: `./a.out -b 2000 data.txt output.txt`

`-d <number of days> <input file> <output file>`

All the customers with a balance due within the number of days specified. Customers with zero balance should not be reported.
Example: `./a.out -d 5 data.txt output.txt`

`-A <input file> <output file>`

Display all the records with the average balance and the total of all balances

All data must be stored in the output file in a nicely formatted table.

Display error messages if any of the following occurs:

- The number of options is incorrect (show a usage message)
- The wrong option is used (show a usage message)
- Any of the files fail to open

Required Functions:

- `bool isValidOption(string option);`
  returns true if the option is valid, false otherwise

-
  ```
  void processBalance(string option, double balance,
                      ifstream &inStream, ofstream &outStream);
  ```
  Reads from the input stream and outputs a report to the output stream based on the option provided (`-a`, `-b`).

-
  ```
  void processBalanceDue(int numDays,
                      ifstream &inStream, ofstream &outStream);
  ```
  Reads from the input stream and outputs a report to the output stream (option `-d`).

- `void outputSummary(ifstream &inStream, ofstream &outStream);`
  Output all the customer' data, average balance, and total of all balances (option `-A`).

Parsing command line arguments

In C++ you can input data (strings) into your program on the command line (command line arguments). You can capture these data by adding two parameters to your main program as follows:

`int main (int argc, char *argv[])`

`argc`: (argument count) number of arguments on the line, including the name of the program

`argv`: (argument vector) list of c-style strings that represent all the arguments including the name of the program (two dimensional array of characters).

For example, executing the command:

```
./a.out -A data.txt output.txt
```

Assigns:

```
argc = 3
argv[0] will "./a.out"
argv[1] will be "-A"
argv[2] will be "data.txt"
argv[3] will be "output.txt"
```

To capture the `argv` data items in your program, just assign them to strings as follows:

```
string option = argv[1];
string inputFileName = argv[2];
string outputFileName = argv[3];
```

You can convert a string to a an int or double using the functions `stoi` and `stod` (section 8.4 of your textbook). You may have to include the compiler option `"-std=c++11"`. For example:

```
double x = stod("23.45"); // assigns 23.45 to x
```

Grading:

Programs that contain syntax errors will earn zero points.
Programs that do not include functions other than main will also earn zero points.
Programs that use global variables other than constants will earn zero points.

Your grade will be determine using the following criteria:

- Correctness: the program works as requested above (35 points).
  - (7 points) isValidOption function
  - (7 Points) processBalance function
  - (7 points) processBalanceDue function
  - (7 points) outputSummary function
  - (7 points) error detection

- **Documentation and Style (5 points)**

Follow the coding style outline on GitHub:
https://github.com/nasseef/cs2400/blob/master/docs/coding-style.md